NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A MODEL OF AN INTEGRATED AIR DEFENSE SYSTEM (IADS) FOR THE TACOPS PROGRAM

by

Bryan F. Smith

September, 1991

Thesis Advisor: LCDR William J. Walsh

Approved for public release; distribution is unlimited.

T258622

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b. OFFICE SYMBOL OR
6c. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000	7a. NAME OF MONITORING ORGANIZATION 7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION	8b. OFFICE SYMBOL 9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

11. TITLE (Including Security Classification)
A MODEL OF AN INTEGRATED AIR DEFENSE SYSTEM (IADS) FOR THE TACOPS PROGRAM

12 PERSONAL AUTHOR(S)
SMITH, Bryan F.

13 TYPE OF REPORT	13b. TIME COVERED FROM TO	14. DATE OF REPORT (Year, Month, Day)	15. Page Count
Master's thesis		1991, September	291

16. SUPPLEMENTAL NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and Identify by block number)
FIELD	GROUP	SUB-GROUP	Integrated Air Defense System, IADS, TACOPS, Computer Model.

19. ABSTRACT (Continue on reverse if necessary and Identify by block number)

The author develops a model and computer code for some of the functions of an Integrated Air Defense System (IADS). These are combined with an existing FORTRAN program, called TACOPS, which is used by the Naval Weapons Center (NWC). The functions and attributes of an IADS such as target detection, track data processing, target position and accuracy estimation, target assignment, information relay to other units and information relay delays are described. Shortcomings of the fire control and coordination capabilities of TACOPS are described. Using the IADS model, the results against a sample engagement are compared to the original TACOPS. Documentation of the code is provided to allow future operators the ability to refine and/or restructure the algorithms as needed. Approaches for improved target engagement coordination and allocation are described. Selection criteria for picking a Measurement of Effectiveness (MOE) are presented.

20 DISTRIBUTION/AVAILABILTIY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC	1a. REPORT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL William J. Walsh	22b. TELEPHONE (Include Area Code) (408)646-3113 22c. OFFICE SYMBOL OR/Wa

Approved for public release; distribution is unlimited.

A Model of an Integrated Air Defense System (IADS)
for the TACOPS program
by

Bryan F. Smith
US Department of Defense
B.S.E.E., University of Washington , 1973

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September, 1991

ABSTRACT

The author develops a model and computer code for some of the functions of an Integrated Air Defense System (IADS). These are combined with an existing FORTRAN program, called TACOPS, which is used by the Naval Weapons Center (NWC). The functions and attributes of an IADS such as target detection, track data processing, target position and accuracy estimation, target assignment, information relay to other units and information relay delays are described. Shortcomings of the fire control and coordination capabilities of TACOPS are described. Using the IADS model, the results against a sample engagement are compared to the original TACOPS. Documentation of the code is provided to allow future operators the ability to refine and/or restructure the algorithms as needed. Selection criteria for picking a Measurement Of Effectiveness (MOE) are presented.

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these porgrams without additional verification is at the risk of the user.

TABLE OF CONTENTS

I. INTRODUCTION 1

A. GOAL OF THESIS 1

B. TACOPS OVERVIEW 1

C. TACOPS DEFICIENCIES IN SAM FIRE CONTROL AND COORDINATION 2

D. PURPOSE AND SCOPE OF THIS THESIS 3

II. TACOPS PROGRAM 4

A. ASSUMPTIONS OF THE PROGRAM 4

1. Discrete Time/Event Step, Expected Value, Low Detail Level 4

2. M-on-N Simulation 6

3. Types of Attackers, Air Defense Units and Targets 7

B. FIRE CONTROL AND COORDINATION 9

1. Detection of an Attacker 9

a. Radar Cross Section 9

b. Masking by Terrain or Jamming 10

c. Launch Envelope of a SAM 11
d. When an Attacker Maneuvers 12
e. Engagement Conditions for a SAM 12
2. Engagement of an Attacker by a SAM unit 13
a. Launch, Flyout, Kill Assess, Reacquisition Delays 13
3. Existing Fire Control/Coordination 14
a. Fire Control for Each SAM unit 14
b. Coordination 15
c. Effectiveness of present control/coordination 15

III. THE INTEGRATED AIR DEFENSE SYSTEM 17
A. Structure 18
1. Purpose 18
2. Function 19
B. Components 20
1. Sensors 20
a. Radars 21
b. Electro-optic and Infrared Devices 23
c. ESM Receivers 24
2. Data Fusion and Decision Centers 25
a. Data Fusion Nodes 25
b. Decision Nodes 25

3. Communications links 26

a. Radio 26

b. Landline/hardwire 27

C. Summary 28

IV. DEGRADING THE PERFORMANCE OF THE IADS 29

A. THE IADS AS AN INFORMATION PROCESSION SYSTEM . . 29

B. SENSORS 32

1. Radars 32

2. Others 33

C. COMMUNICATIONS LINKS 34

D. DATA FUSION and DECISION NODES 35

1. Data Fusion Node 37

2. Decision Nodes 38

E. SUMMARY 40

V. IADS ALGORITHMS 42

A. Modeling Considerations 42

B. THE IADS NETWORK 43

1. Connections in the Network 43

a. IADS Types 43

b. Maximum number of IADS Elements 44

c. Defining the IADS Connections using Directed Graph .. 44
d. Controlling the ultimate destination of a message 48
2. Message Content, Delay and Transfer 49
a. Message Content 49
b. Message Delay 50
c. Message Holding Array 52
d. Message Transfer 52
C. IADS Modeled Functions and Elements 55
1. EW Site 55
2. EW Reporting Station 57
3. ADWOC/Filter Center 58
4. Battalion Center 59
5. SAM Units 60

VI. A SAMPLE SCENARIO 61
A. PURPOSE 61
B. THE ASSUMPTIONS 61
C. THE SCENARIO 61
1. The IADS Network 62
a. Specifying the IADS 63
b. Specifying the IADS Network Connections 64
c. Non-SAM targets i.d. numbers for the IADS elements. .. 66

2. The Physical Layout 67
3. Picking the EW Site Delay Time 70
D. RESULTS 72

VII. PROGRAM DOCUMENTATION 75
A. IADS SUBROUTINES 75
B. DOCUMENTATION IN THE IADS SUBROUTINES 75
C. MEANING OF THE 'INCLUDE' STATEMENT 77
D. CREATING THE INPUT DATA FILE 77
E. RUNNING TACOPS WITH AND WITHOUT THE IADS 78
F. DEBUG OPTIONS 79

VIII. MOEs OF THE IADS 81
A. ATTRITION AGAINST THE ATTACKERS 81
1. Maximum Total Damage to the Attackers 81
B. ATTRITION OF THE IADS-DEFENDED TARGETS 82
C. INTERRUPTION OF IADS RELATED FUNCTIONS 82
1. Number of SAMs launched 83
2. Delay in Transferring Messages 83
D. CONCLUSION 84

IX. CONCLUSIONS 85

A. TACOPS DOCUMENTATION 85
B. THE INTEGRATED AIR DEFENSE SYSTEM 85
C. IADS ALGORITHMS 85
D. SUMMARY 86

LIST OF REFERENCES 88

BIBLIOGRAPHY 89

APPENDIX A 90

APPENDIX B 99

APPENDIX C 118

APPENDIX D 127

APPENDIX E 135

INITIAL DISTRIBUTION LIST 279

I. INTRODUCTION

A. GOAL OF THESIS

The goal of this thesis is to develop computer algorithms which model the Early Warning (EW) sensors, network connections and information transfer in an Integrated Air Defense System (IADS). These algorithms will be incorporated with an existing program called "TACTics and OPtionS" (TACOPS) to enable greater realism of this anti-air warfare simulation.

B. TACOPS OVERVIEW

TACOPS is a FORTRAN program originally written over 10 years ago at the Naval Weapons Center (NWC), China Lake, California. It is a many-versus-many (m-on-n) anti-air warfare (AAW) simulation involving aircraft, air to surface and cruise missiles (attackers) attempting to penetrate a defensive area to hit assigned targets. The area has air defense units that simulate Surface to Air Missile (SAM) systems which will attempt to shoot down the incoming attackers. The program produces as an output the expected level of damage done to the targets, attackers and SAM sites. TACOPS is run as a deterministic discrete event model which includes emitter locations, attacker flightpaths, a variety of attacker weapons and multiple SAM site engagements. TACOPS is currently being used at the NWC in weapon system

effectiveness studies and is also continually being modified as needed by the users at NWC to accommodate new systems.

C. TACOPS DEFICIENCIES IN SAM FIRE CONTROL AND COORDINATION

TACOPS currently has one major drawback - the absence of substantial coordination in the allocation of the SAM units to the attackers. The resulting non-optimal allocation of SAM's to attackers is manifest in that some attackers will have more SAMs shoot at them than is needed to reach a desired probability of kill (Pk), while other attackers may not be shot at all. The attackers' effectiveness against their assigned targets depends upon the attackers' probability of survival (Ps) which is based upon the SAMs failure to destroy them (1-Pk). As a consequence, the attackers' effectiveness as a system results partially from ineffective assignment of the air defense assets. Because of this it is difficult to accurately assess the effectiveness of the SAM systems.

The current model assumes that detection and location of all attackers is performed by some unspecified system. This information is available to all the SAM units instantly. The systems that do the detection, location and transfer of information are not able to be targeted and damaged or destroyed. Both of these characteristics of the TACOPS model are inaccurate. SAMs will not have instantaneous target data, and parts of the self-defense system are usually high priority targets of attacking units. By allowing targetable system elements and real

control of SAMs and detection emitters, the air defense options can be better investigated using different weapons or tactics.

D. PURPOSE AND SCOPE OF THIS THESIS

The major functions of a general IADS will be defined. Consistent with the assumptions and precision of modeling in TACOPS, those IADS functions which will produce the most impact on the effectiveness of the SAM systems will be modeled and coded to fit into TACOPS. The intent is to create modifications and subroutines that allow these critical functions of an IADS - detection, location, information transfer and delays, to be incorporated in TACOPS. The subroutines presented will give a generic IADS capability. The user will tailor these subroutines to meet his specific modeling needs. The formulation of the model and the code will be documented with the goal that a future user with a specific set of needs, functions or algorithms in mind will have the structure and knowledge to use this work to implement that specific simulation using parts of these algorithms. The IADS model will be used in a sample engagement and the results compared to TACOPS without the IADS algorithms. The last section will include a description of the advantages and disadvantages in picking various Methods of Effectiveness (MOEs).

II. TACOPS PROGRAM

There are several reasons for providing a description of the TACOPS model. One is to give the reader an understanding of the existing model. This will help in understanding how the IADS algorithms were implemented within the original code. Another is to document my understanding of the functioning and assumptions of TACOPS. This will ensure that the premises from which I redesigned the model are current and consistent with the understanding of the current uses at NWC. A final reason is to provide some documentation on the model, since there presently exists very little documentation on TACOPS.

A. ASSUMPTIONS OF THE PROGRAM

1. Discrete Time/Event Step, Expected Value, Low Detail Level

TACOPS is a hybrid of discrete time step and event step models. In a discrete event step model, the time when a significant event will occur is computed. Then the simulation clock is moved forward in the increment required to be at the time when the event occurs. The time of the next event is then computed and so on until the end of the simulation. In a time step model, time is increased in constant intervals (usually 1 second) and, after each time step, the program determines what events occur (i.e. update positions of attackers). The hybrid model used by TACOPS combines the characteristics of both types. The time steps are constant and discrete (usually at 1 second

increments). After each time increment, the position of the attackers is updated (time step) and a check is made to see if any predetermined significant events (i.e. a SAM shoots a missile) are to occur (event step). Most of the time steps will accomplish little more than updating the position of all the attackers. The increment of time advancement is small enough so that the program will not advance too far to exclude any of the predetermined events.

The program uses an expected value approach in computing the probabilities of kill (Pk) and probabilities of survival (Ps) during engagements. This means that a fixed and constant value will be used in the single shot Pk computations for all engagements whether it be the first or tenth engagement with this same attacker. An average value of Pk or Ps will be used to represent the effectiveness of the system for all conditions. The overall Ps of an attacker or target will change according to the number of engagements. TACOPS is also a deterministic model in that no random events occur. The results of one run will be identical with all subsequent runs of the same scenario.

The model does account for a variety of events and functions for each of the systems simulated, but the level of detail is low. Examples of this detail level are: 1) An attacker will move from one three dimensional (X,Y,Z) position to the next by specifying these positions (called checkpoints) in the input data file. The program moves the attacker along a straight line at constant speed between checkpoints and instantaneously changes the course of the attacker when is reaches its next checkpoint. The attacker is always assumed to be

oriented straight and level; its body axis is parallel and aligned along the line connecting the last and the next checkpoint, and the pitch, roll and yaw of the body are zero. 2) Weapons launched by penetrating aircraft are assumed to always fly to their correct impact point without any inflight system malfunction or any terminal targeting errors. 3) The Pk of a SAM system against an attacker is the same anywhere inside its launch envelope.

2. M-on-N Simulation

TACOPS is a m-on-n AAW model. To understand how this differs from other AAW models, an overview of those model types are presented. There are general classes of AAW models which are differentiated by the number of units engaging each other and the way the engagements are modeled. The smallest number of engaging units is one SAM unit against one attacker (called a one-on-one) model. Generally these models will be designed with a high level of detail about each system. When the specific number of units gets very large, the specific interactions of individual units are not modeled but the overall force levels and attrition are typically computed using a Lanchester type equation. This is called a force level model. In between these two types of models is a model whose number of individual units are constrained enough to allow the computation of the specific results of individual units. This is the many-on-many model.

An m-on-n model is the appropriate model when the air defense is dense enough that the individual units coverage of an area overlap to allow

multiple simultaneous engagements of single or multiple attacker(s). This type of model must be used in the m-on-n simulation because the results of a one-on-one engagement cannot be extrapolated to the m-on-n situation. The two types of models are not completely independent. The averaged results of the one-on-one model are used in the simulated engagements of the m-on-n model. Some details from the one-on-one model are lost in the averaging but the m-on-n model does allow the simulation of more than one attacker against more than one system. Any unusual results of one engagement are expected to be smoothed out when the results of the multiple engagements of the complete simulation are produced.

3. Types of Attackers, Air Defense Units and Targets

TACOPS assumes that three different types of attackers can exist in the simulation. They are functionally grouped and can be thought of as 1) aircraft, 2) fire and forget air to surface missiles, and 3) missiles needing post launch control. These three different type of attackers share many common attributes.

* All can be (and normally are) shot at by the SAM units. Each attacker is an air vehicle of some sort (aircraft, glide bomb, cruise missile).
* All aircraft start and end in midair (takeoff and landing at an airfield is not simulated).
* All missiles must be launched by an aircraft in the simulation (they cannot just appear).

* All attackers have their motion defined by checkpoints in an input data file. Any motion other than going straight through the checkpoints is not allowed.

* All attackers will follow the flightpaths until they reach their last checkpoint (for missiles, this is their impact point) or their Ps is below the entered threshold and they are presumed dead and taken out of the simulation.

Although each attacker type is different in some capabilities (i.e. flightpath, speed), there are other capabilities that are identical for all attackers of the same type. One example of this would be the Pk of the SAMs against the attacker type.

The air defense units also have similarities and differences. All units are SAM systems with up to 9 unique types. The unique types will have different launch envelopes, velocity of its missile, location in the model, etc. The different types will have identical fire control logic.

There exists a class of objects in the model which are called 'non-SAM' targets. These simulate any ground target which has no capability to shoot back or be offensive in any way. They are the primary targets for which the attackers objective is to penetrate through the air defense network to strike. Other scenarios might have the SAM units as the primary targets of the attackers and the 'non-SAM' targets not even engaged.

B. FIRE CONTROL AND COORDINATION

1. Detection of an Attacker

The first step in engaging an attacker is to detect it. This will provide location information which will be used in determining if the attacker is or will be within launch range of any SAM units.

a. Radar Cross Section

As stated previously, an overall assumption in the model is that radar will be used for EW detection and SAM unit tracking. To determine if an attacker is engageable, a simple formula is used in a subroutine which represents the radar range equation. It determines whether the attacker in question would have sufficient signal strength for the radar to detect it. Thus each attacker has a number in the input data file that is associated with its radar cross section and a number associated with the parameters of the radar in the detection process. These variables are named CS(*) and BS(*) respectively. With the nominal values used at NWC, the attackers will be detected at a range which is typically much larger than the maximum launch range of any SAM units launch envelope. At present the detection formula has little effect upon SAM engageability. It could be useful in the future if the effect of low cross section on attacker effectiveness is desired.

b. Masking by Terrain or Jamming

Masking interferes with the radars ability to detect and track the attacker. To jam a radar, a jammer aircraft emits noise which competes with the attackers reflected radar signal. If this noise is of sufficient strength, the attacker will not be detected. The radar range equation predicts when the attacker is close enough to the radar being jammed so that the reflected signal of the attacker is stronger than the jamming noise and again is detectable. This is simulated in the program by adjusting the maximum range of the detection envelope. If the jamming is effective in stopping detection, the modified detection envelope will be smaller than the theoretical envelope range of the jammed detection radar.

The term terrain masking refers to the situation where some natural obstacle is directly between the radar and the attacker blocking the direct path needed by the radar signals to proceed to and return from the attacker. This occurs when an attacker flies behind a mountain. Although the program allows each SAM site and attacker to have three dimensional (3D) coordinates, no high resolution computations are made which would simulate specific 3D terrain masking. A general assumption is made that terrain masking will exist for low elevation angle engagements. The terrain is characterized by two numbers in the input data file which simulate a barrier fence of a given height at a given range. This fence completely encloses the SAM unit in a 360 degree circle. If the attacker is beyond the range of the barrier fence, the

attackers elevation angle must be above the terrain masking angle limit for the attackers to be detected. If the attackers range is less than the range to the barrier fence, the attacker can be detected regardless of its altitude. (Imagine sitting in the center of a gigantic saucer and trying to observe objects moving both within the saucer and beyond its lip.)

If the attacker had already been detected and was being tracked when the attacker is terrain masked, tracking may be lost. If the terrain masking is temporary within a time limit defined by the user, the program simulates that the radar will not lose track. Beyond this time limit the track is completely lost on that attacker by the affected unit.

c. Launch Envelope of a SAM

The launch envelope is a bounded volume. If the attacker is within the bounds of the envelope of a given SAM unit, it can engaged by the SAM. The extent of the launch envelope for each different type of SAM system is originally found in the input data file. This exists as a pair of range and altitude points which define the perimeter of the envelope. The TACOPS program uses this two dimensional (2D) perimeter and creates a 3D volume by computing the 2D perimeter rotated about the vertical (Z) axis. The program uses this 3D volume as the maximum enclosure in which attackers may be engaged.

d. When an Attacker Maneuvers

Whenever an attacker initially enters the simulation, the time of the intersections of its flightpath with all SAM units' launch envelopes is computed. This is based on the extrapolation of the current course of the attacker through all possible launch envelopes in the simulation. When an attacker reaches its next checkpoint, its speed and course may change. If either one of these parameters change, it will change the positions where and times when the attacker flies through the SAM units' launch envelopes. These intersection points are recomputed whenever any attacker reaches a checkpoint.

e. Engagement Conditions for a SAM

In addition to all the above conditions, each SAM unit will have some conditions specific to that unit that must exist before it will be allowed to fire against an attacker. The SAM unit must be 'alive', that is, its probability of survival at this time must be above that set as the minimum survival probability for all SAM units. If the SAM unit's Ps is below the minimum threshold, the program will ignore this SAM unit for the remainder of the simulation. The SAM unit also must have at least one missile to shoot otherwise it will not be selected to engage any attackers.

2. Engagement of an Attacker by a SAM unit

a. Launch, Flyout, Kill Assess, Reacquisition Delays

The simulation starts with all attackers at a range in excess of all launch envelopes. As an attacker approaches the time when it will enter the launch envelope, TACOPS alerts the SAM site. Each SAM type will have a unique track time which is the time it takes to accomplish the functions of turning on its radar and then locating and tracking the attacker with enough precision to allow a missile to be launched. The program adds the track time to the time when the attacker will enter the envelope to get a 'start radiating' time for that SAM site. When the simulation clock time reaches this 'start radiating' time, the SAM unit is given the attacker identification (I.D.) number and set from a 'non-radiating' mode to a 'target acquisition' mode. When the simulation clock reaches the time when the attacker enters the launch envelope, the SAM units' mode is set to 'launched SAM'. The flyout time of the SAM is computed as the flight time it will take for the SAM to intercept the attacker given its current flightpath and velocity. The simulation will wait until the simulation clock has reached this intercept time and then determine if successful intercept conditions still exist to warrant Pk computations of this intercept. If the attacker is still within the launch envelope, the Pk of the missile is applied to the Ps of the attacker and a new Ps is computed. The general form would be:

$$P_s=(1-P_k)^{SHOTS}$$

After the intercept, the SAM waits a certain time before being allowed to continue to the next step in the engagement. This time delay represents the kill assess time. This is the time it would take for this SAM type unit to assess if the attacker has been 'killed'. If the attacker has been killed, the simulation will determine if this unit can engage any other attackers. If no other attackers are engageable at this time, the unit is set to a 'non-radiating' mode. If another attacker is already within its envelope, the units mode is set to 'start radiating' and the engagement sequence repeats for that attacker. If the original attacker was not killed and the unit has more missiles to fire, another SAM is fired at the attacker. All previously mentioned computations are done and the mode of the unit is again set to 'launched SAM'. This one-on-one engagement continues until the unit has no more missiles to fire or is killed (by weapons launched at it), the attacker is killed or it exits the launch envelope.

3. Existing Fire Control/Coordination

a. Fire Control for Each SAM unit

The input data file contains numeric values which are used by the program for a type of fire control. The data consists of a binary number for each SAM unit and every attacker. The value indicates which attackers each

SAM unit will be allowed to engage. A value of 1 allows engagement while a 0 value allows no possible engagement of the attacker by that SAM unit. This allows absolute control of possible engagements but puts the operator is the precarious position of dictating before the simulation of the battle which engagements will be allowed. This is usually known after the simulation has completed execution and is used to confine the simulation in order to stop some unrealistic simulated engagements. This ends up 'gaming' the simulation.

A few lines of code in TACOPS do presently accomplish a level of fire control logic. One line in the subroutine SHOT1 is used when a SAM unit has to choose between more than one attacker to engage. The code will cause the attacker with the highest Ps to be engaged.

b. Coordination

One input in the data file is used to select between what is called 'free-fire' and 'one-on-one only' fire control logic. In the 'one-on-one only' mode, the program will allow a SAM unit to engage an attacker only if that attacker is not being engaged by another unit. In the 'free-fire' mode, each unit engages attackers independently. No consideration is given to the number of other units that may also be simultaneously engaging this attacker.

c. Effectiveness of present control/coordination

The code that selects the highest Ps attacker when faced with a choice of which to engage focuses on only the present moment. The EW

sensors may know of a much higher Ps attacker that will come within the units launch envelope in a second or two, but the SAM site does not consider this.

The 'free-fire' mode can result in a lead attacker being engaged by multiple units simultaneously and allowing trailing and nearby attackers to pass through the launch envelopes without being engaged. This would be highly unrealistic. The 'one-on-one only' mode will not allow the error of the 'free-fire' mode to occur but limits the units per attacker to only one. More units that could engage are not allowed to do so resulting in a overall loss in air defense effectiveness.

III. THE INTEGRATED AIR DEFENSE SYSTEM

Most military organizations have air defense systems but their composition varies greatly. At the lowest range of effectiveness, the system would consist of hand-held SAMs and World War II era anti-aircraft artillery (AAA). At the highest, the system would be composed of the most sophisticated weapons systems connected by state-of-the-art electronics. In this chapter, the concept, composition and function of a typical air defense system and its elements are presented. The system components are differentiated by type and function. The description will be presented with an slant towards their incorporation in the IADS model.

The majority of the information in this chapter and the following chapter was obtained from Lt. Thomas J. Bernota's Masters of Science thesis (Bernota, 1990). That information is contained in this thesis to allow this document to be complete. Many sources describe specific components or functions of the IADS. His thesis is an excellent single source for a description of the multiple attributes of an IADS. It also contains a large listing of references. Both were applicable to this thesis. Some areas of his thesis have been condensed, reworded or modified so that the information better applies to the topic of this thesis. (Lt. Bernota's thesis focus was jamming of the IADS while this thesis focus is modeling it.)

A. Structure

1. Purpose

To model an air defense system, it is necessary to thoroughly understand two areas: (1) the purpose of the defense system; and (2) the method by which it achieves its purpose. The ultimate purpose of the air defense system is to prevent any attacker from reaching and damaging its target. While perfect protection is generally not possible, the system might have a lesser goal. This goal could be to induce a high enough attrition on the attackers that either 1) the enemy does not attack at all; or 2) the cost of the attack is so high that these defended targets are not attacked again. Although specific missions of an air defense system are usually grouped under the categories of strategic or tactical employment, this thesis does not make any distinctions.

A purpose in employing the IADS model could be to evaluate different methods (stealth technology, approach tactics, jamming, selectively damaging parts of the IADS) which prevent the air defense from being effective. Specifically, an air defense system must be able to establish and maintain a track on an attacker to function at all. This track must be sufficiently accurate and complete to direct a weapon against the attacker. Therefore, the timeliness and accuracy of the attackers track information are one type of measure as to how well the defense system works. Other measures are: 1) the amount of damage done to the attackers; and 2) the amount of damage the attackers inflict upon the air defense system and the ground targets the air defense was tasked to protect.

2. Function

The components of an IADS can be physically far apart. For an IADS to achieve the general mission objective, some means of communication and coordination between the various components must exist. These components will be described in more detail.

When a command, control and communications (C^3) process is used with the linked components, the system is more effective than the individual components acting alone. This is called a force multiplier effect. Modern aircraft and weapons fly at high speeds thus requiring the IADS to function in a timely manner if it is to successfully perform its mission. A way to defeat an IADS is to delay or prevent the completion of those tasks. Included in the tasks the C^3 system must support are (Fowler and Nesbit, 1980):

* positioning of units to maintain a continuous air defense umbrella over moving ground forces;
* distribution of preplanned friendly air support operation data (safe corridors, altitudes, ingress and egress time, etc.);
* real time identification of friend or foe;
* coordination of fighter and SAM air defenses;
* coordination of target assignments to prevent unnecessary missile expenditures;
* direct fire control of both SAMs and interceptor aircraft;
* distribution of warnings or tipoffs from other intelligence sources.

Without being properly designed and operated, the numerous tasks the C^3 system is required to handle could negate the force-multiplier effect. Damaging

the components and disrupting or delaying the functions of the components would also negate the force-multiplier effect.

The physical layout of the IADS is influenced by a variety of factors. These are terrain masking limits, importance of the targets to be protected, and SAM/interceptor aircraft capability and availability. For maximum effectiveness, the sensors would be placed at expected ingress routes. Similarly, weapons would be placed around vital assets and/or along expected strike flightpaths. The location of the components making up the communications net would also be influenced by terrain masking and may be influenced by the maneuverability of the connected elements.

B. Components

The components of an IADS are broadly categorized as follows (Fitts, 1980, pp. 209-237; Schleher, 1986, p.342):

* sensors: radars, electro-optic devices, electronic support measures equipment, human observers;
* data fusion and command centers: reporting and filter centers, command posts at all levels (division, brigade, battalion, etc.)
* weapons: surface to air missiles, anti-aircraft artillery, interceptor aircraft;
* communications: voice radio, data links, landlines.

1. Sensors

IADS sensors include the components of the C^3 structure used to obtain localizing positional information on any attackers. Other sensors may detect the

presence of attackers without giving location information. Although not as useful as positional information, this knowledge can still be put to good use.

a. Radars

A radar is any device that uses the electromagnetic spectrum to obtain position information on another object. Radars may be categorized by the functions they perform. For our purposes, they are grouped as follows:

* Early warning (EW),
* Target acquisition (TA),
* Height finders (HF),
* Target trackers (TT),
* Multifunction.

Early warning radars are used to provide the first notification of the approach of attackers. The information is used to notify the defense. Depending on the capabilities of the particular radar, the information could also be used to determine the number and position of the attackers. These radars are usually high powered to produce the long range detection capability required to fulfill their mission. They tend to provide azimuth and range information only but some have a limited height-finding capability (i.e. 3D). Two dimensional information alone is not sufficient to use for target tracking. Also, the information from an EW radar also usually lacks the accuracy needed to develop a solution to pass to a target tracker. Because the scan rates of most long range search radars are low (5-10 revolutions per

minute (rpm)), and detection is more difficult at extended ranges, a confirmed detection tends to take more time than with other radars.

The target acquisition (TA) radar is also a long range radar and is generally associated with a weapon control or target track radar. As its name implies, the TA radar attempts to develop information that is accurate enough to slew a target tracker onto a target, or in some cases, the weapon itself. The TA radar may not be capable of determining the altitude of the aircraft by itself. In that case, it will operate in conjunction with a height-finding radar. The TA radar can either be an independent radar or an operating mode of a multifunction radar. Thus used in conjunction with an EW radar, this combination will be able to detect and locate the attackers sufficiently for target tracking purposes.

The height-finder (HF) provides altitude information on attackers using two dimensional surveillance radars. A typical HF radar slews in azimuth to the bearing obtained by the surveillance radar, and then scans in elevation to acquire the attacker. Once detected, the combination of azimuth, elevation and range is sufficient to compute the altitude information. The altitude of the attacker, in conjunction with the azimuth and range obtained by the 2D radar, is sufficient and accurate enough for use by a the target tracker. That would be the next step. (Schleher, 1986, p.231)

The target tracker (TT), or weapon control radar, provides sufficiently accurate information that allows the weapon to engage and destroy the target. This type of radar is associated with SAMs or AAA. The design and range of this type of radar will vary depending upon the weapon system it supports but will usually have

the shortest range of all other radar types. Some of the factors influencing the capability of the TT radar include the weapons range and coverage, volume, and type of weapon (e.g. point or area defense) (Schleher, 1986, p.233). Target tracker radars influence the use and effectiveness of the terminal threats (SAMs,AAA). These threat's capabilities directly affect the probability of survival of the attackers. Therefore, MOEs associated with delaying the acquisition and tracking information pertinent to terminal threat performance are of interest.

Previous to this section, each function described was associated with a unique group of equipment. Advanced radars have been able to combine multiple functions described previously into one system. The multifunction radar will perform several of the above roles. A single radar system may be able to provide target acquisition, altitude, and tracking data, as well as serve as a weapons guidance transmitter. Detection may still need to be accomplished by an EW radar. The implication is the elimination of a need to transfer target data between separate radars and data communication links. Thus, the targeting process is quicker, simpler and more reliable. Phased array radars are extremely capable and can operate and scan quickly. They are typically the radar of choice in multifunction systems.

b. Electro-optic and Infrared Devices

Passive electro-optic (EO) and infrared (IR) devices may exist with a radar. If they do exist, they are usually subordinate to a primary radar. They have limited range capability (on the order of tens of miles or less) and generally operate only in fair or clear air mass weather conditions. There are EO devices that are used

to track targets, and in some applications, to search for targets. These devices can give very accurate bearing information on the target. Their narrow field of view make them impractical when searching for targets that do not have some initial bearing data. The acquisition device provides input to the IADS, and the tracker acts much as a radar target tracker.

c. ESM Receivers

Electronic support measures (ESM) receivers are used to search, intercept, locate and identify sources of electromagnetic radiation. The ESM function is used for threat recognition. As the name implies, they are passive elements that receive electromagnetic spectrum and bearing information from a transmitter. By correlating the signals with enemy emitters, target identification is possible. Using bearings from multiple ESM receiver locations, rough attacker location is possible. The main advantage of these elements is that they function passively without revealing their location. In a low observable mode, the IADS would only use the ESM receivers to search for attackers. If the ingressing attackers are radiating any emitters (i.e. jamming), an assessment of degradation of ground-based ESM and direction-finding stations should be made. This assessment will be used by the central control if the IADS to determine if additional elements of the IADS must be employed to get the needed information of the attackers.

2. Data Fusion and Decision Centers

Centers in the IADS that are the focal points for the reception of information from sensor sources are defined as data fusion nodes. The elements of the IADS responsible for using that information and acting on it are defined as decision nodes.

a. Data Fusion Nodes

Data fusion nodes are centers that collect and process information. There are two types of nodes which differ by the type of information and IADS elements they handle. The radar reporting stations are associated with the data from an early warning radar. The radar reporting station compiles a picture of the air situation from the outlying EW radars. Its outputs are to the filter center. The filter center correlates this and also receives inputs from ESM stations and any other sources (ground- or air-based observation posts). The outputs of the filter center can be modeled to include general alerts to the IADS and data on the air situation to the weapons control center.

b. Decision Nodes

All command centers are considered to be decision centers. There are different levels of command centers. The high echelon command centers are those from the division level upwards. They are primarily responsible for planning the allocation of forces on a daily cycle and do not handle threats of an immediate

nature. Disrupting the command and control of such a command center would not have an immediate impact on the enemy presently engaged in an attack.

In contrast, lower echelon command centers offer a more viable target for disruption or destruction.

> Regiment and battalion command centers are assigned to execute the orders of the higher echelons. They are primarily involved in the "control" of command and control. They...conduct the real time coordination of air support, ...air defense, etc. (Fowler, 1980, pp. 44-45)

Timely countermeasures or destruction against these command centers could disrupt the process required for successful air defense weapons applications.

The command posts that function as air defense weapons operations centers (ADWOC) are included in this category. They are alternatively know as air defense direction centers or weapons operations centers. This is where allocations of the weapons covering the entire air defense network and involving current attacks are determined.

3. Communications links

In the previous sections, components of the IADS structure used to obtain and utilize information were reviewed. Communications connects those components with each other and allows the information to be shared. The types of communication links may be broadly categorized as radio or landline/hardwire.

a. Radio

Radio communications operate in many areas of the electromagnetic spectrum. Radios generally operate in the frequency range from 1 to 500 Mhz

(Megahertz). Microwave and satellite communications may operate in the 1 to 12 GHz (Gigahertz) region.

The type of data to be transmitted may influence the type of transmission used. The oldest and most commonly known broadcast communication means is analog in nature. Analog transmission is more applicable for the representation of sound and visual information (which is continuous in nature). For modern C^3 systems, digital communication is beginning to dominate over analog forms. Digital techniques lend themselves well to the transmission of text and numbers.

There are different communications radios which are dependant upon the frequency in which they operate. High frequency (HF) radio waves propagate beyond the horizon, and can either hug the ground or be bent by the ionosphere. Very High Frequency (VHF) and Ultra High Frequency (UHF) radio waves essentially travel via line-of site. If the nodes in the communications net are physically beyond direct line-of-site, VHF and UHF links will use relay stations between the primary nodes to allow them to function.

b. Landline/hardwire

A large volume of military communications is transmitted by telephonic or telegraphic hardwire connections. These connections would be made of electrical cable or fiber optic lines that are laid on or under ground or strung like standard telephone lines. Landlines are used for voice, data and video information.

A primary benefit to the user is the relative difficulty of destroying such a link. Although susceptible to physical destruction, they are impervious to jamming.

For short ranges and fixed positions, landlines are easily established. They are not suitable where long-distance connections must be established rapidly or when connections are between mobile command posts. Additionally, a large volume of telephonic and telegraphic communication is often transmitted by radio relay at some point between connection nodes.

> It is safe to say that potentially every long distance military communication will be carried over part of its path by electromagnetic radiation. (Fitts, 1980, p.129)

Jamming and damaging the relay nodes can disrupt the information transfer.

C. Summary

In this chapter, the major functional components of the IADS were examined. Each plays a vital role in the ability of the system to achieve the mission of the IADS. Each function must be part of an IADS model. In the next chapter, the vulnerabilities of the system and its components will be presented.

IV. DEGRADING THE PERFORMANCE OF THE IADS

Modeling the IADS requires understanding how it operates, both when it is in perfect operating order and under degraded (i.e. elements are damaged/destroyed or jammed) conditions. The previous chapter covered the functions and components of the IADS. In this section, the interaction of those components, and some ideas on degrading the system are presented. Then a discussion of the methods in which to defeat the individual components is provided along with the potential impacts on the overall system.

A. THE IADS AS AN INFORMATION PROCESSION SYSTEM

The approach used in Lt. Bernota's thesis is that the IADS can be viewed as an information processing system.

> Basically, an air defense system is an information processing system. It does have physical resources - men, guns, missiles and aircraft - which it controls, but its control over these resources is dependent upon its processing of information about the threat. (Fitts, 1980, pp. 228-237)

Associated with this approach are four defects which reduce the value of the information. They are, in increasing order of effect on the system, error, bias, distortion, and delay. (Fitts, 1980, p. 229) The first three defects typically relate to the positional information of the attacker, while delay is related to the transfer of the information regardless of the quality of the data.

Positional error of the attacker will arise as part of the natural noise induced error present in a radar system. It can arise from azimuth inaccuracies in the EW radar which are present when the system is operating at its peak or when damage is induced from a strike. Jamming induces error due to the error added to the system. The system attempts to remove error by smoothing or averaging observations so that the random fluctuations cancel out over time. Electronic counter measures (ECM) only becomes effective if the averaging time becomes so long that the result is a delay in estimating the correct position of the target.

Bias is the tendency of the system to produce a measurement that consistently deviates from the actual measurement. Repeated underestimates of the bearing angle of the incoming raid indicate a bias in the system.

Distortion may already exist in the system or may be externally increased.

> Distortion in an air defense system may arise because of the limits of its equipment. Slower scan rates on the radars, or extended processing time of its data correlation centers, may result in a renewed picture of the air situation only once every 10 seconds. High attacker speeds, in conjunction with turns, may deviate route projections from the actual position in that period of time. ECM enhances the distortion effect by allowing fewer glimpses of the actual position of the aircraft, or increases the processing demands through the introduction of false targets. (Bernota, 1990, p. 23)

Thus what the attackers are actually doing is distorted by the lack of accurate information by the air defense system resulting in degraded performance of the IADS.

The last information defect, delay, usually has the most serious consequences. The effects of other defects may also be grouped under the category of time delay.

Recall how error and distortion will affect the IADS and how the IADS handles these. The IADS must deal with these defects until the solution is within acceptable bounds before the system will be able to bring weapons on the attacker. In this way, error and distortion may be viewed by the amount of delay they introduce. TACOPS does not have a high-resolution detection algorithm nor does it use the estimate of the exact position of the aircraft in its engagement and terminal effects algorithms. Modeling the specific defects that positional error, bias and distortion cause is not compatible with TACOPS modeling assumptions. Their manifestation will be modeled by the delay they induce. This thesis concentrates on the delays in the IADS.

Timeliness has utility. The IADS command and control may eventually be able to resolve the position of the attacker, given enough time. If high speed attackers are to be engaged before they can do any damage, timing is critical. If the position solution is obtained as a result of the attention drawn to the attacker when it delivers bombs on its target, or as the attacker egress out of the effective weapons range of the SAMs, the information comes too late to have any chance to preclude damage or to employ surface to air weapons optimally. If the attackers have already exited from the SAM envelopes, the information will not be useful at all. The three key areas where delays may be introduced into the system correspond to the components of the IADS described in Chapter III. Two direct methods are used to cause delays in the IADS components; physical damage and jamming. Direct methods against

communications links result in denying or delaying the system ability to move any information between nodes.

Additional delay may occur as an indirect result of the effectiveness of the direct methods. This delay would occur in the decision nodes.

> The...effect is that the commander will often delay a decision because his information is uncertain. The propensity for a commander to so act is obviously a very human quality but the pressure is still there, and the resultant delay might have serious consequences. (Fitts, 1980, p. 230)

Some of the means of introducing the delays into each of these components and how to treat them in modeling them are now discussed.

B. SENSORS

Mission planning aids currently in use are capable of determining the radio frequency (RF) propagation restrictions due to the terrain (called terrain masking). TACOPS already crudely determines terrain masking so this section does not expand on this phenomenon. However, because hiding from the sensors is an important means of denying information to the IADS, the user will need to be aware of its effects.

1. Radars

Destruction is the ultimate combat tool. (Atchision, 1987, p. 66) Destruction of a radar results in an "infinite delay" for that sensor. The term "infinite" is used to describe a delay that extends the arrival of the information beyond the time period of usefulness. If destruction produces too long a delay, then it is assumed that a small level of damage to a radar will produce a corresponding lower delay.

The IADS model allows the equating of delay to damage level. Destruction may not be preferable in all cases. Use of Anti-Radiation Missiles (ARMs) that contribute to the attrition of the air defense system's radars is desirable, but not feasible for all sensors. Some elements of the IADS may be too heavily defended to attack. In this case, jamming may be the weapon of choice. Jamming may be modeled as a denial of information (infinite delay) until the aircraft is close enough (burn through range) so that the radar can clearly discern the target. Lt. Bernota's thesis gives a detailed description of the radar range equation and the detection performance of a radar in a jamming environment.

2. Others

> Where operational requirements include any radiation from the attacker, the vulnerability to passive ESM stations must be determined. As was the case with radars, detection of emissions is a function of the signal-to-noise ratio (SNR) set in the receiving equipment. The detection probability and utility of information vary considerably with different types of ESM equipment. (Bernota, 1990, p. 32)

The detection event is a random event. Like modeling the radar for TACOPS, the user must also come up with deterministic value to model the ESM equipment.

EO/IR devices also have their deficiencies.

> EO/IR detectors are limited by several factors. Like a radar, they may have a limited field of view. They may be dependent on external inputs to slew to the target, or they may be capable of a search mode. The primary range restriction other than line-of-sight is due to atmospheric attenuation. (Bernota, 1990, p. 32)

The prime advantage of these devices is the highly accurate bearing-to-target information they can provide. Their main disadvantage is that their narrow field of view results in a very long search time unless given an small search area.

C. COMMUNICATIONS LINKS

Degrading communications links is more difficult than degrading the sensors (Bernota, 1990). Two possible methods of degradation are considered; damage and jamming. The previous reference goes into some detail as the differences between radars and communications links and their signal equations.

Actual damage of the links was not modeled. The algorithms are capable of being easily modified to allow the links between nodes to be modeled as working perfectly or being completely broken. Modifications to allow marginally functional links would require more work. The user will have to evaluate vulnerability of the actual hardware to model this component.

Encryption of the signals makes jamming less effective. There is not a good consensus on what constitutes effective communications jamming because of the complexity and diversity of communications methods used. In this IADS model, the relationships between link disruption and intelligence denial and between intelligence denial and operational significance is difficult to quantify.

On the one hand, unless the disruption is 100% effective, the intended message or an equivalent operational clue may be transferred.

> An "alert" may be inadvertently by conveyed by the jamming itself. (Van Brunt, 1978, p. 141)

If an attack has not yet been detected and was not anticipated, the start of jamming would indicate to the IADS that hostilities were soon to commence. Detection and location of the attackers would not be required before the IADS would be at a more heightened state of readiness. On the other hand, even if complete obscuration of the signal is not achievable, there may still be substantial benefits to using less "successful" jamming. If the jamming imposes a finite delay on message traffic at several points along the communications path, the cumulative effect could be significant. This, along with delays imposed by actual sensor jamming, may result in sufficient confusion to delay or prevent successful missile engagement of the attacker. (Bernota, 1990, p. 36) This would have a major influence on one of the primary MOEs in TACOPS.

D. DATA FUSION and DECISION NODES

Data fusion nodes were defined here as those centers that receive and consolidate information from the IADS' sensors. Decision nodes are those that use the information, or direct other sensors to obtain further information. Specifically, we are referring to the ADWOC and different levels of command posts.

This next section focuses on the interactions between the sensors of the system and their control elements. In the model, each node may be connected to a set of other nodes. The resulting network may be described by the set of connections from each node, and the set of connections to each node. A network with connections in this format is called a directed graph.

To facilitate development of the model, some simplifying assumptions regarding how delays are imposed on the nodes are made. A collection of information will be referred to as a message or package.

> It was previously mentioned that a commander may delay making a decision if his information is incomplete. It is assumed that each decision node is restricted in the action it may pursue by the completeness of the information package to which it has access. The two general cases then are access to incomplete information, and access to complete information. A subset of both cases is that the information is no longer current, and therefore unusable. (Bernota, 1990, p. 38)

It is further assumed that gaining and processing information will take time. The time delay will be a constant value when the IADS component is undamaged and will correspondingly increase as the components' damage level increases.

Each level of decision node has specific requirements for the type and amount of information. As an example, a target tracker radar (as part of a battalion level command post) may require a message which contains information on target azimuth and altitude. With that information, it may slew to a small enough region so that target acquisition is possible in a reasonably short amount of time. If the information it has is incomplete or not accurate enough, it may not be directed to radiate at all because to do so would increase its probability of detection and ARMs could be used against it. Alternatively, it may be directed to attempt an autonomous acquisition, with a long expected delay time.

The requirements for all the levels of decision nodes in all situations are too long to list here. However, the following sections look at some possible information requirements of the IADS' nodes, and the restrictions placed on their operation by

incomplete packages. The model has been created with the flexibility to accommodate a variety of connections and user defined control conditions.

1. Data Fusion Node

A data fusion node acts as an information traffic direction center, relaying messages, consolidating inputs, and filtering extraneous information. This information is then passed to any receiving nodes it is connected to. The sample scenario presented here has two data fusion nodes: the EW reporting station and the Filter Center. The inputs to the EW reporting station come from the EW and HF radars (assumed to be part of the EW site). From the information gained by using these radars, a complete information package (message) would consist of a three-dimensional position report on the attackers each EW site can detect. This would be sufficient to allow computations for target allocation assignments. Incomplete packages would lack any or all of the three dimensions. An incomplete package may just consist of existence of jamming and its azimuth direction of arrival and still be somewhat useful. In either case, there is no restriction on attempting to pass that information to the Filter Center. Also from the EW Reporting Station, (if the ADWOC is destroyed and will not alert the battalion centers and SAM units) an Air Situation Broadcast ("ALERT") is made to let the rest of the system know of the impending strike. In general, the alert may contain limited sector information, or just consist of a general alert. In the sample scenario, the entire IADS will try to autonomously get detection and track on the attackers.

The Filter Center receives, in the sample scenario, the information from the EW site and a redundant copy from the EW reporting station. Note that the information is passed via one of several possible parallel communications links. Eliminating one of the links would not prohibit the message from its destination node. As previously mentioned, there may be an additional delay in reverting to a secondary parallel link because each node will incur a delay in handling the message. The Filter Center combines the information received from all EW sites to produce a overall description of the total attackers detected at each moment.

2. Decision Nodes

As a decision node, the ADWOC commander uses the overall picture of the information on all the attackers to assign specific SAM units to engage the attackers. If it is necessary, the ADWOC may attempt to obtain missing information, or refine the attackers position. This could be done by using its TA or HF radar or coordinating this information with other reports.

> Three possible situations that may occur are as follows. No delays are incurred at the ADWOC if it is a complete package. If the package is incomplete, a delay is imposed until the information is obtained, at which time the ADWOC commander may designate which battalion or battery will be responsible for engaging the attacker. Alternatively, a delay is imposed until the ADWOC commander chooses to instruct the battalion to assume responsibility for detection and engagement of the attacker. (Bernota, 1990, p.45)

Assume the ADWOC, knowing the position of the attacker, directs Battalion 1 to engage the attacker. After any delays incurred have elapsed, the Battalion Command Post (CP) receives the instructions to prosecute the target. With sufficient information for target acquisition, the Battalion CP attempts to direct SAM

Batteries under his command to track and fire upon the aircraft. If the position of the attacker were not known accurately, the Battalion Center would attempt to get the lacking information, by using a TA radar. Without the exact location of the attacker, the SAM sites would have do their own detection, acquisition, identification and tracking of the attacker before they could engage the attacker. This is normally a lengthy process. Assigning the SAM unit to do all these functions would not be as effective as a functioning IADS with complete information.

In the TACOPS model, target trackers require current down range, cross range and altitude (x,y,z) position data to acquire the attacker. In actual systems, azimuth and altitude data will usually be sufficient for acquisition. Attacker position data may be rendered obsolete by terrain masking, delayed jamming, or the attacker exiting the envelope of the radar. If the communication link to a battery is completed, but the data passed is not current, the target tracking radar cannot find the attacker without a prolonged search. As long as the radar radiates, the attackers ESM equipment can locate this radar. Since that increases the vulnerability to ARMs, the battery CP would probably decide to use his own TA radar to obtain the information to feed his TT radar. This would reduce the chance of damage to the only components of the IADS that can engage the attacker. If the TA radar is being jammed, the CP must decide on using the TT radar for autonomous acquisition.

The above situations illustrate the wide number of conditions that must be considered for a strike against an IADS protected target. Modeling an actual

IADS would require detailed knowledge of the decision process. However, from the descriptions, the following requirements need to be part of the model:

* a matrix of all communication links, defining who can send message to whom;
* a logical decision process as to what events require sending messages and the ultimate destination of that message;
* a definition of what data needs to be contained in the message;
* a decision process as to the radiating state of the IADS elements, the conditions when this state should change;
* provisions to duplicate operations of a destroyed ADWOC and the SAM units trying autonomous operations;
* a decision process for target allocation and SAM engagements.

In the sample scenario, the Filter Center and the ADWOC are assumed to be synonymous and collocated. Once the overall detection information is assembled (the Filter Center's function), the model allows the SAM sites to engage the attackers (the ADWOC's function).

A complete description of the techniques used in and capabilities of the model are presented in the next chapter.

E. SUMMARY

The IADS can be thought of as an information processing system. Delay is one of the significant errors associated with the IADS and will be focused on in this thesis. Delays are incurred in three areas of the IADS: sensors, communications, and decision and data fusion nodes. Each category has unique considerations as to how the information flow delay is incurred. The cumulative effect is a delay in obtaining

a track on the attackers and the resulting loss of effectiveness in the IADS area defense role.

V. IADS ALGORITHMS

Now that the functions of the IADS and its elements are defined, algorithms can be written to represent these functions in a simulation. This chapter will describe four areas. First, the assumptions and simplifications that were adopted are described. Next, the data file inputs which define the IADS network and connections are specified, and the response of the information transfer of the IADS network is presented. Finally, the algorithms developed to simulate the IADS elements are detailed. Some areas will be highly detailed so that the user can take these insights and tailor them to his future needs.

A. Modeling Considerations

One of the major considerations in developing these algorithms was to allow the IADS model to be compatible to the original TACOPS program. This influenced the choices in level of detail, modeling assumptions and the programming language.

Since the original TACOPS is written in the FORTRAN77 computer programming language, that necessitated the use of that same language in writing the IADS algorithms. (Object oriented languages such as MODSIM II would be better structured to model all these algorithms.) The original TACOPS could run on a desktop personal computer (PC). Again to keep the IADS model compatible with the TACOPS, algorithm coding was conducted which would keep memory requirements within the size available in a PC.

Although TACOPS has the capability to be used in a Monte-Carlo fashion, it has never (to my knowledge) been successfully used that way. It has always been used in its expected value deterministic mode. Hence, the IADS algorithms were written assuming an expected value mode would be used.

TACOPS computes times when future events occur similar to an event step model. It increments the simulation clock by one time increment similar to a time step model and waits until the time the event is to occur equals the simulation time. The IADS algorithms uses the same approach to be compatible with the original program.

B. THE IADS NETWORK

1. Connections in the Network

a. IADS Types

The model assumes that the major IADS functions are differentiated into five different components which in this document will be called IADS 'types'. A unique number distinguishes each type. The numbers associated with the different 'types' are; 1 - an EW site, 2 - an EW Reporting Station, 3 - the ADWOC center, 4 - a Battalion Center and 5 - a SAM unit. These types communicate their information in a hierarchy that is illustrated in Figure 1.

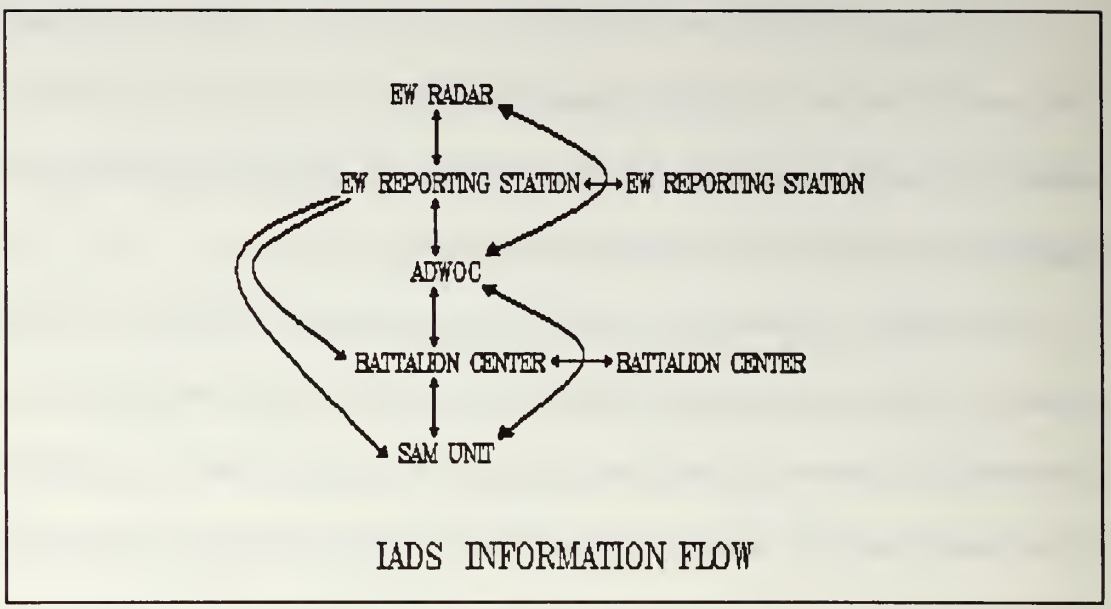

Figure 1 - Information Flow in the IADS

b. Maximum number of IADS Elements

Assumptions are also made as to the maximum number of 'elements' for each of these types. (This defines the array sizes.) There can each be a maximum of 9 EW sites, EW Reporting Stations and Battalion Centers. There can be only one ADWOC element. From the previous original TACOPS assumptions, the maximum number of SAM elements (units) is set by the parameter 'JM' in the original code.

c. Defining the IADS Connections using Directed Graph

The concept of a 'directed graph' is used to envision the connections between elements in the IADS network. Each element in the network can send messages to other elements in the network only when it is connected to that element. A connection does not indicate that message traffic is allowed to travel both ways. A directed graph specifies that messages may only be sent in a specified direction.

See Figure 2 which illustrates one IADS element with a directed connection to another element.

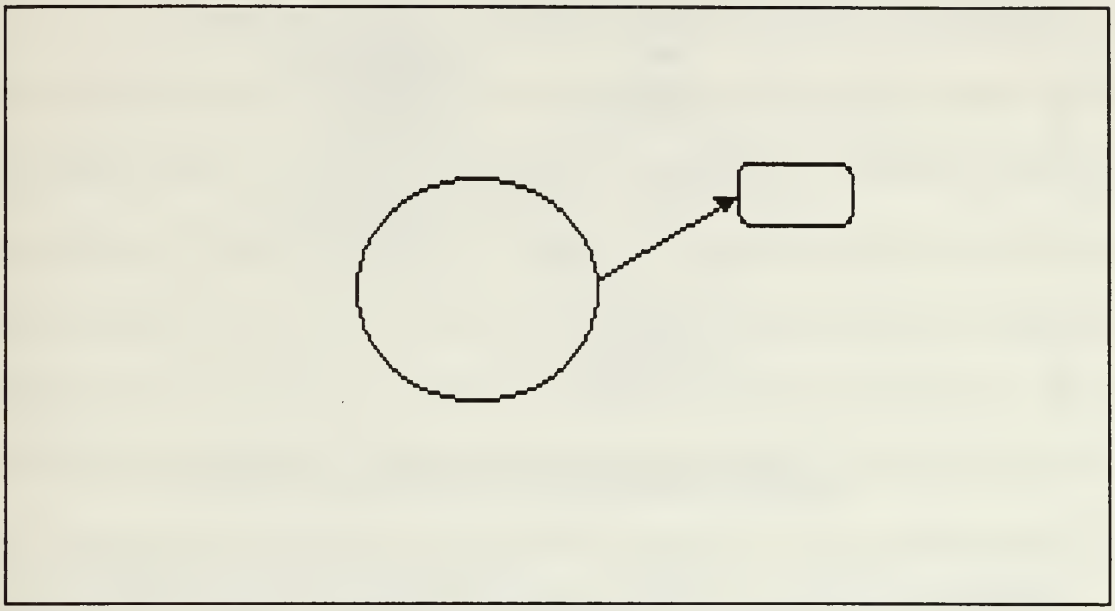

Figure 2 - Example of Directed Graph Connection

All the IADS connections are assigned a four number designator. In each single four number packet, the connection and the direction from one element to another is established. The format is as follows: Isolate a pair of elements in the IADS network and visualize the appropriate direction of message flow in the connection. The first pair of numbers represent the originating IADS type number and the element number of the IADS element in question. The last two numbers specify the IADS type number and the element number of the receiving IADS element. (Example, 1,2,2,4 means that type '1' (EW site), element '2' (site #2), can send a message to type '2' (EW Reporting Station), element '4' (Station #4).) See

Figure 3. To completely specify the IADS, all elements must be considered individually and all their connections directing from each element must be listed.

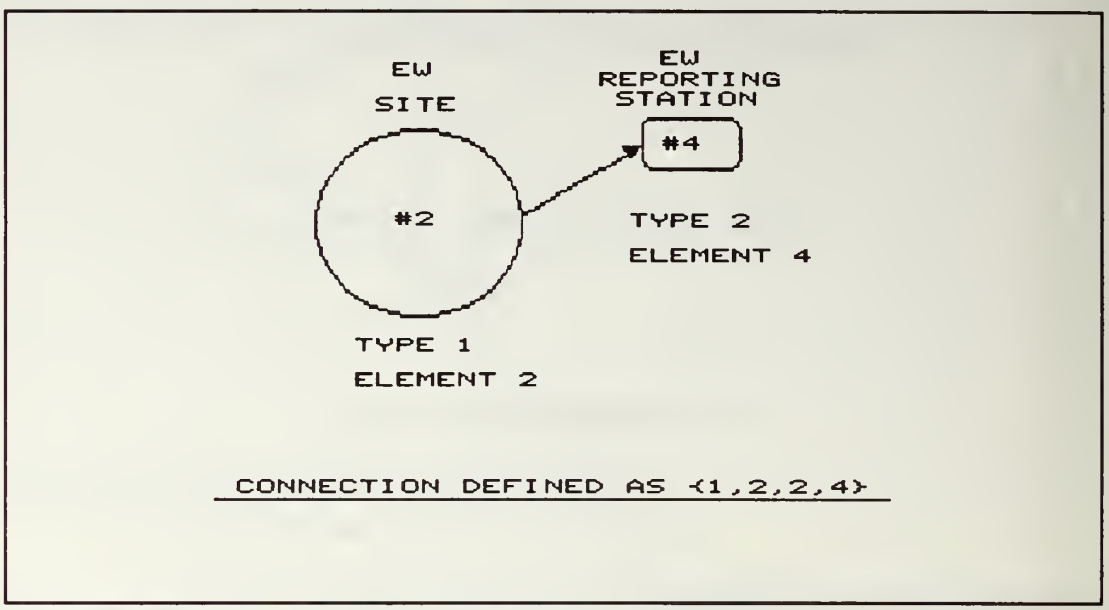

Figure 3 - Example of a Connections Data set

There is a possible source of confusion in determining the EW or SAM site element number. This is because there are two numbering schemes being used to associate identification numbers to EW and SAM sites. Confusion may arise because the SAM sites use the same numbering scheme for damage assessment and specifying its IADS connection element number while the EW sites use two different numbering schemes for these areas. The differences are now detailed.

The original TACOPS code identifies all SAM sites in the input data block using a site identification number. This site number comes from the order of occurrence of that site's data in the site data block (i.e. the first site in the data block is #1, the next is #2, etc.). This identification number will be used for the SAM site's

element number. This scheme will not be used for the EW sites element number. In defining the IADS connections, another numbering scheme is used.

The model associates a unique number of all EW sites numbering the sites starting with #1 and proceeding sequentially up to the maximum number of EW sites in the scenario. The element numbering association for an EW site is linked to the relative position of that EW site's data in the data block to the other EW sites. In the input data file, the EW and SAM site's positional data are all grouped together in a block. There is no relative ordering sequence in this block for the EW or SAM sites data. (The EW sites may be listed all together and then all the SAM sites or they may be randomly interspersed.) Starting at the top of this block, each successive line signifies a different site. Whatever the order in this block, the occurrence of the first EW site will be designated EW site element #1. The next EW site occurring in the block will be EW site element #2 even if there are multiple SAM sites between the EW sites. The user must correctly identify the EW site's element number using this scheme. He must use this element number when he specifies the IADS connection. Here is an example.

If there are 2 EW sites and 4 SAM sites, there are 6 sites total. Assume that the EW sites were put in the input data file as the first and third lines of the SAM data block. Thus the original TACOPS site identification numbers for the EW sites would be 1 and 3 and the SAM sites would be numbered 2,4,5 and 6. When referring to IADS types and elements numberings, the type '1' IADS elements would be for EW sites only regardless of original site number. For these two EW

sites, the EW site element numbers #1 and #2 would refer to the original TACOPS site numbers 1 and 3.

d. Controlling the ultimate destination of a message

Although the network can have each element connected to many other elements, it is not desired to have messages follow down all possible paths. Loops are created when a IADS network is setup which allows messages to follow along multiple parallel paths. This allows the possibility that messages may be passed along forever by going over the same loop over and over again. See Figure 4 for an example of multiple parallel paths and loops. This situation would slow down the running of the program by having unneeded copies of messages existing and moving through the network and require too much memory (array size). Two techniques were created to handle this problem. One technique uses the source and final destination information in the message to selectively direct messages along specific paths from a node in the network. The other technique keeps track of each unique message.

By using the source and/or final destination information in the received message, the user can put logical statements in the program to control the direction of messages. This will result in messages being passed along specified paths from a node in the network. The model uses this technique in all subroutines that send messages. A form was created to accompany this part of the model and can be found useful in organizing the logic flow of the code. The copy of this form is in APPENDIX A. Users should find this helpful in understanding the logic flow of these

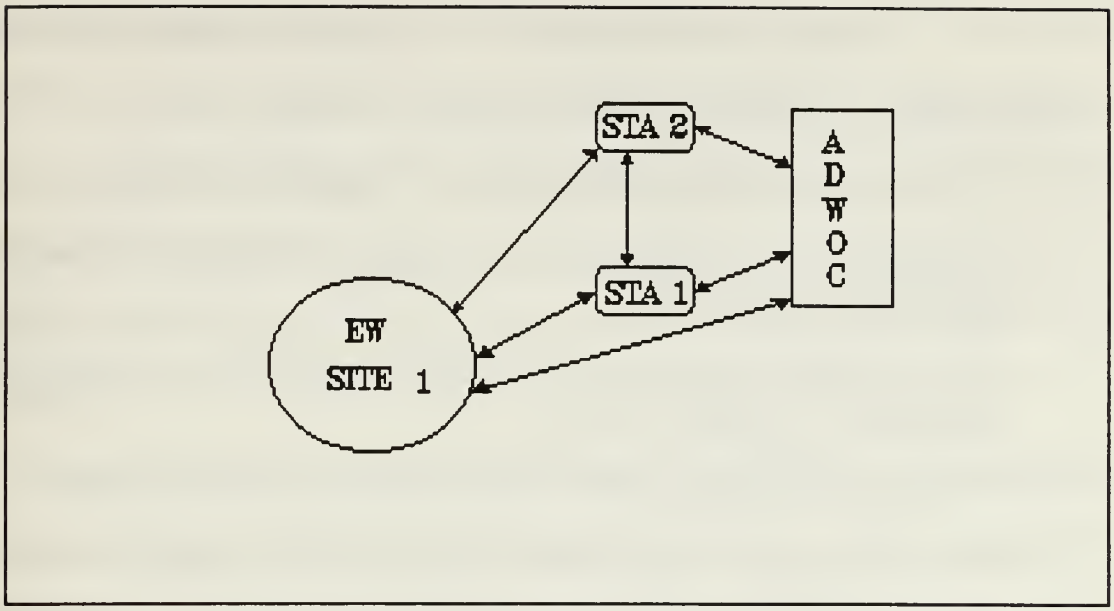

Figure 4 - Multiple Parallel Paths and Loops in the IADS Network

subroutines and in any future modifications or expansions of this code along similar lines.

2. Message Content, Delay and Transfer

a. Message Content

The IADS model and the associated network presently requires 13 pieces of information to function. These pieces of information are contained in a 'message'. If the model is expanded, it will probably require more pieces of information per message. To account for this, the size of the array which holds the message is assigned by a PARAMETER statement. This allows this upper limit to be easily modified if more pieces of information are needed.

Each of the 13 data values in a message has a unique meaning based upon its position in the message. The meaning of each data value is:

1. The time in the simulation when the event triggering this message occurred;
2. The time in the simulation when this message is planned to be sent to the next IADS network node;
3. The probability of survival of the originating IADS element when the event triggering this message occurred;
4. The present IADS type number from which this message is relayed;
5. The present IADS element number from which this message is relayed;
6. The IADS type number for the receiver of this relay;
7. The IADS element number for the receiver of this relay;
8. The 'action' bit: a value associated with an action that the receiving element must perform;
9. This message's identification number;
10. This message's originating IADS type number;
11. This message's originating IADS element number;
12. This message's final destination IADS type number;
13. This message's final destination IADS element number.

b. Message Delay

Messages are not sent immediately upon occurrence of the triggering event. The data for the message will be assembled and put in a message holding array until it is ready for transmission. The gap between the time of the event and the time when the message is transmitted to the next IADS network node is called the delay

time. It is the sum of two delays; normal processing and damage related delays. Each IADS element will have in the input data file numbers specifying these values.

A normal delay will be a fixed number associated with the time required for the IADS element to process the message and transfer it to the next node. This would be the minimum delay associated with the element if that element was not damaged.

A delay due to the sustained damage of the IADS element is dependant upon preassigned values in the input data file. They are used in the subroutine 'CDELAY' to compute the extra delay for any damage inflicted upon the element. The realistic relationship between damage level and delay was unknown so the subroutine has a linear and an exponential equation to compute the delay. Both equations are present to allow the user a choice in the computation of the delay function. The user must determine which delay equation is appropriate. Delay in both equations is accomplished by correlating the element's damage to the probability of survival. The input values in the data file specify the maximum delay that can be computed when the elements Ps goes to 0.0 and the increment of delay to be computed for the increment of Ps. The equations are simple and the relationship of the input value to the computed delay are easy to assess once studied.

Once both delays are computed, they are summed. This combined delay is the time delay the message must wait from the occurrence of the event until the message is transmitted. Adding this delay to the time of the event goes in position 2 of the message described above.

c. Message Holding Array

As previously mentioned, information is gathered and formed into a message at the time an event occurs. All messages generated are inserted into a message holding array (called MESHOLD). At each simulation clock increment, the holding array is checked to determine if a message is ready for transmission. This is accomplished by checking the message ready time with the current simulation clock time. If the ready time is within one 'DT' (the variable name of the simulation time increment, usually one second), the message is transmitted. The message is then removed from the array. By removing transmitted messages, the memory size of the array needed to handle all messages during the run of the simulation is kept to a minimum. (The message removal is done by subroutine SHIFTU).

When a weapon impacts and damages an IADS element, the model scans through the message holding array to determine if any messages are waiting for transmission from the newly damaged element. If so, the corresponding message's ready time is increased to account for this new damage level. If the IADS element is damaged severely enough to be considered destroyed (below its destruction threshold), all messages from element are deleted.

d. Message Transfer

When a message is transmitted from one element to another, the message is removed from the message holding array. The message is now considered residing at the receiving IADS element. The receiving element may send the message to other IADS elements. The elements which will get a copy of this message is based upon

the IADS connections and the users defined routing of messages. This will use the origin and destination IADS type and element numbers previously mentioned.

One might envision the message being created at one of the IADS elements, waiting until it is ready to be transmitted. When ready, it and any other copies of it instantly move to the appropriate adjacent network connections. The message waits at these elements until they (independently) are ready to transmit the received message.

Two conditions can stop the message from moving along the network. The first condition is when the message arrives at its final destination. The data is then processed but not passed along to any further elements. The data in this message may illicit other actions requiring messages, but they will not be copies of the original message.

The second condition which will stop a message from moving along the network is when an IADS element refuses to accept the message. There are two ways for a message to be refused. One is if the IADS element is destroyed and therefore not functioning. Once an element is destroyed, the model does not allow the element to send or receive any messages. It does not stop any other IADS elements from trying to send messages to this element. The other means of stopping receipt of a message is if a copy of this message has already been received by this IADS element. The model tags each unique message at the point of origin with a message identification (i.d.) number (data #9) and stores this number for each IADS element which relays that message. When an element is notified by the model that a message

is going to be sent to it, it looks at the message's i.d. number and compares it to all the i.d. numbers previously handled. If there is a match, the message is ignored by not receiving it. This halts the message from being copied and passed along by this IADS element. This technique is used to keep the number of messages existing on the network at any given time to a minimum so that the size of the program arrays will remain small.

If the message is only to be copied and relayed, only four data values in the message are altered. Positions #4 and #5 of the message are changed to be those of the relaying element's value and positions #6 and #7 are changed to be the next IADS type and element values in the network connection. The message to be sent out will now appear to come from the relaying IADS element. The origin and final destination data values are not changed. Thus in each message, the transmitting and receiving IADS elements are identified along with the message origin and final destination. This maintains all the information needed by the model to relay the information to its correct destination.

If this message necessitates the creation of a message with a different meaning than the one received, a new message must be created. The data package is constructed for this new message along with a unique i.d. number and the message is then put on the holding array.

C. IADS Modeled Functions and Elements

As previously mentioned five IADS elements were modeled associating IADS functions to each element. The following is a detailed description of the modeling of each of these IADS elements.

1. EW Site

The purpose of the EW site is to be the first element of the IADS to detect the attackers. It is assumed that the EW site will have multiple (EW, HF, TA) radars. These will produce positional information on the attacker with sufficient accuracy to be useful in a target allocation algorithm. Thus 'detection' of an attacker indicates that its position is known accurately enough for the SAM units to engage it.

The three dimensional detection range capability of the EW site is modeled using the original TACOPS algorithms that compute the SAM launch envelope capability. It is assumed that an EW site is a radiating site just like a SAM site. An EW site's capabilities are specified in the same way a SAM sites capabilities are specified in the original TACOPS. Its data is included in the section where the original TACOPS had included the SAM site data. The user inputs other data further on in the data file to differentiate between the SAM and the EW sites.

The model computes the attackers flightpath's intersections with the EW sites three dimensional detection envelope. From these intersections, the entrance and exit times are determined. The original code in the TACOPS subroutine 'SHOT0' (which determines if an attacker is within the SAMs launch envelope) was

copied and simplified for use in the EW detection envelope. If the attacker is within the envelope, the attacker is considered 'detected' by that EW site. If the attacker is outside the bounds of the envelope, the attacker is not 'detected' by the EW site.

The EW detection envelope has other restrictions which allow an attacker not to be detected even if it is within the detection envelope. These are the same as the SAMs limitations in the original TACOPS model. One limitation is terrain masking. Each SAM and EW site has terrain masking limits specified which will not allow the radar to detect or track attackers that are hidden by terrain. Another limitation is jamming. The effects of jamming are not presently modeled within the EW site detection algorithm. The similarities in the effects of SAM site and EW site jamming would allow modifications of the EW detection algorithm to incorporate the procedure in which SAM sites are jammed.

The algorithm for the EW site detection assumes that the TACOPS simulation will query all EW sites at each simulation clock increment. Changes in the status of the number of detected attackers for each EW site is important. The model keeps track of the detection of each site's attackers. When the EW site is queried as to its present detection status (by calling the subroutine EWDET), the present detection status is compared to the previous detection status for all attackers in the simulation. If a new attacker is detected, a '1' is associated with that EW site and that attacker is placed in a memory array. If a previously detected attacker is not reacquired, a '-1' is associated with that EW site and that attacker. If an attackers detection status has not changed from the previous look, a '0' is put into the array.

If there are any changes in the detection status of any of the attackers for each EW site, a new message must be transmitted. This change in detection status is stored in the message array as defined earlier and processed at its final destination (the ADWOC).

An EW site sends a message by calling the EWMES subroutine. EWMES creates the message in the proper format and places it in a message holding array. The IADS element the message will be transmitted to depends upon the users defined connections of the IADS network and the ultimate destination of the message. It is assumed that the EW information would primarily go the ADWOC. Copies of this information would also be sent along other parallel paths to allow for the IADS functional reliability and redundancy.

2. EW Reporting Station

The EW Reporting Station represents a data fusion node of the IADS. Since it is assumed that the EW site will gather positional information on the attacker, the reporting station will serve the function of a parallel path from the EW site to the ADWOC. It has already been shown how the user can create multiple serial and parallel connections between the ADWOC and the EW sites using these reporting stations.

If the ADWOC is functioning, the reporting station will only need to pass the messages along from the EW sites to the ADWOC. If the ADWOC is not functioning (has been destroyed), the reporting station should initiate a new message, an 'alert', to the other IADS elements and specifically the SAM units that an attack

in imminent. To generate an 'alert' message, a new message is created by the EW reporting station. This 'alert' message will be sent to the connecting network battalion center and SAM units. In this 'alert' message, the 'action' position is set to '1' indicating an alert.

The IADS algorithm presently sets all the SAM elements to the radiating (target track) mode upon receipt of an alert. Since the EW detection range is assumed to be much farther than the SAMs launch range, the alert is typically received well before the attackers arrive within the SAMs envelope. If there is no target to track, the SAMs will shut down after a few seconds. This is part of the original TACOPS algorithm that the IADS model implemented. The user will need to expand the logic and code for this type of message if the SAMs are to remain in a radiating mode.

3. ADWOC/Filter Center

The model assumes that the ADWOC combines the functions of a filter center and a decision node. This is where the major target allocation and IADS control (i.e. who will radiate) decisions will be made. The EW and SAM sites will also direct their information to this central point.

The ADWOC will control the radiating status of all EW and SAM sites. Presently the SAMs firing control logic is the same as that in the original TACOPS model. This means that, given a detection from the EW site, a SAM will radiate only when a target is within its launch envelope. The user will need to develop SAM site radiation control algorithms as part of the improved future target allocation

algorithms suggested later. The ADWOC can presently control the EW sites radiating status by sending a message. This message is sent to the desired site with the 'action' position set to the desired command. A '0' instructs the EW site to stop radiating (and thus stop is detection process). A '1' instructs the EW site to start radiating. The site remains in the commanded state until ordered otherwise or it is destroyed.

The ADWOC performs another function. It gets the combined detection status of all attackers from all the EW sites message information. The model keeps the detection status of each EW site based solely upon the information received from messages. (This mimics the situation where the detection status is based on time delayed data that may not reflect the present conditions.) The ADWOC then processes all information to come up with a value which indicates the total number of EW sites which have detected each attacker. If the value is greater than zero, at least one EW site has detected that attacker. This information is used in the target track algorithm of the SAM units. No SAM unit is allowed to engage an attacker until the ADWOC has received detection information.

4. Battalion Center

The Battalion center in the IADS is assumed to function as a data fusion node. It will act similar to an EW reporting station. It will pass copies of messages between the ADWOC and the SAM units. The user can similarly create multiple parallel paths for these messages to ensure that the entire IADS network has no weak points for information transfer. There is one difference between a battalion center and a EW reporting station. It is presently assumed that there are no events

where a battalion center will be the source of a unique message. (The EW reporting station could generate an 'alert' message.)

5. SAM Units

The SAM units are the only elements in the IADS that have weapons to engage the attackers. All the original algorithms associated with target track through kill assessment in the original TACOPS model were implemented into the IADS model. The subroutines with the name 'SHOT' in the original TACOPS were used and renamed 'FIRE' in the IADS model. The only modification to the code has already been mentioned. Before a SAM unit can engage an attacker, it has to meet the additional requirement that the ADWOC has a notification on the attacker in question. The use of the original SAM algorithm allowed the original TACOPS model to function with the IADS model incorporated into it.

VI. A SAMPLE SCENARIO

A. PURPOSE

There are three main purposes in presenting this sample scenario. One is to document how the user can construct an IADS network. Another is to show how that network is described by inputs to its associated input data file. Finally, this will illustrate the effects of information delay on attacker attrition in the modified TACOPS model.

B. THE ASSUMPTIONS

The measures of effectiveness will be the amount of attrition done to the attackers by the SAMs. SAMs will use free-fire engagement control logic. The only parameter that will vary will be the attackers approach angle. This will indicate any sensitivity to the approach angle for this scenario. For each approach angle, the original TACOPS and the IADS TACOPS will compute results and these will be compared.

C. THE SCENARIO

Specifying the inputs for the input data file is not straightforward. With the explanations below, the chapter on program documentation and APPENDIX B, the user should will be able to produce the necessary inputs.

1. The IADS Network

Two EW sites, two EW reporting stations and two Battalion centers along with one ADWOC and four SAM sites define this sample network. The desired information flow connections of this network is shown in Figure 5. The most important connections are between the EW sites and the ADWOC. This direct connection means that the EW sites' message will get to the ADWOC with the EW sites' message time delay as the only delay. The multiple parallel paths will take longer than this direct connection. The delay in this detection information will be a critical parameter affecting the attacker attrition. This is because this detection information must arrive at the ADWOC before the SAMs are allowed to engage the attackers.

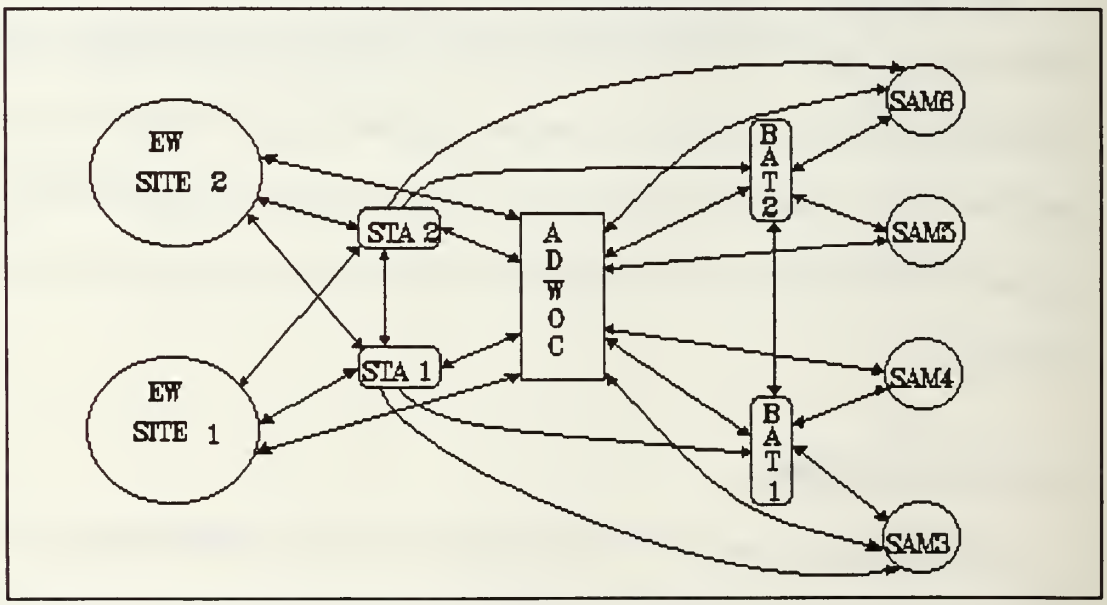

Figure 5 - IADS Network for Sample Scenario

a. Specifying the IADS

In addition to the original information needed by TACOPS for its input data file, data specifying all the parameters of the IADS are needed. A copy of the data file used in this scenario is listed in APPENDIX C. To produce an input data file associated with this sample network, the user starts by indicating that this run will use the IADS model versus the original TACOPS model. This is done by having the third value in the first line of data be greater than or equal to 5. This is highlighted as 'LABEL A' in APPENDIX C. The exact value specifies the fire control logic used by the SAMs. In this case, 5 indicates a free-firing logic with IADS. A 0 indicates a free-firing logic without IADS. See the chapter on program documentation for a list of all values and their associated meaning in fire control logic.

Next the total number of EW sites in this scenario is specified. This and other IADS inputs are located after the original TACOPS data is specified. This line is highlighted as 'LABEL B' in APPENDIX C. Since two EW sites are present, the value '2' is input. The succeeding values indicate which of the sites in the site data block are EW versus SAM sites. The model uses this to identify and differentiate the EW sites from the SAM sites. Next, the normal and damage delay parameters are specified for each EW site. This is highlighted as 'LABEL C'. In our scenario, we will vary the normal delay time to make the IADS sluggish and note the effects. Next, the minimum probability of survival is specified for all EW site types is specified. Each EW sites Ps must be at or above this value to stay in the

simulation. Otherwise the site is considered 'dead' and is no longer included in the simulation.

A pattern for inputting the data for the other IADS types exists. It is identical to the preceding description of the EW site data except it does not include the EW/SAM i.d. numbers. This pattern is; 1) state the total number of elements for this IADS type, 2) list the time delay parameters for each element and 3) specify the minimum Ps for this IADS type. The model expects the data to be in order of the type number (i.e. 2,3,4,5). This signifies that the values for EW Reporting Stations be specified next, followed by the values for the ADWOC, followed by the values for the Battalion Centers and finally the values for the SAM elements.

Two exceptions to this pattern exist. One relates to the ADWOC data line for the minimum Ps. There is an additional value on the same line following the minimum Ps value. This is highlighted as 'LABEL D'. This specifies the initial Ps of the ADWOC. Specifying this value below the minimum Ps has the effect of removing the ADWOC before the simulation starts. This allows the user to experiment with an IADS without the command and control provided by the ADWOC. The second exception is in the SAM area. This is highlighted as 'LABEL E'. The total number of SAM units in the scenario and the minimum Ps are not specified here. It has been previously specified in the original TACOPS data.

b. Specifying the IADS Network Connections

Next, the user identifies each one of the IADS elements by type and element number. (The convention for specifying the IADS components by type is

now described.) Starting at the far left of Figure 5, two EW sites are located. EW sites are IADS type '1'. The IADS element numbers for these EW sites are #1 and #2. Proceeding to the right are two EW Reporting Stations. These are IADS type '2' and are given station element numbers of #1 and #2. Then comes the ADWOC which is type '3' and is #1. Continuing to the right, two Battalion centers exist in this network. They are type '4' and are elements #1 and #2. Finally, the SAM sites are type '5' in the IADS network. The input data file block defines the EW and SAM sites parameters ('LABEL H'). Because of their relative location in this block, the four SAM sites are associated with the original TACOPS SAM target identification numbers #3, #4, #5 and #6.

Now that all the parts of the IADS has been identified, the numbers associated with the directed graph connections for the entire network are specified. (The model does not require any specific order in specifying the connections but a systematic approach will probably reduce errors. Start with all the EW sites, then specify all the EW Reporting Station connections, then the ADWOC, the Battalion Center and finally all the SAM site connections.) Starting with IADS EW sites (type '1'); element '1' has a connection to the (type '2') EW Reporting Station #1. This first connection is specified by the data set {1,1,2,1}. This area is highlighted as 'LABEL F' in APPENDIX C. The next connection from the same EW site is to EW Reporting Station #2. This is specified by the data set {1,1,2,2}. The last connection from this EW site is to the ADWOC (type '3'). Even though there is only one ADWOC, the model expects the #1 as an element number. This set would be

{1,1,3,1}. These three lines are found in the sample data file in APPENDIX C. The rest of the network is similarly specified.

c. Non-SAM targets i.d. numbers for the IADS elements.

If the IADS elements are going to be attacked, they must be assigned non-SAM target identifier numbers so their attrition can be computed. Assigning this type of identifier to the IADS element lets the TACOPS model know that this elements attrition will be handled differently then the SAM (radiating and possibly shooting) elements in the simulation. This structure of SAM and non-SAM type targets comes from the original TACOPS and was used in the IADS model as well. If an IADS element is not going to be attacked, it can but does not need to be assigned a target identification number.

These identifying numbers are placed after the IADS connection block. This area is highlighted as 'LABEL G' in APPENDIX C. Three values in each line form the data set. The first is the non-SAM target number, next is the IADS type number and finally the IADS element number. The order in specifying target numbers is up to the user. The model does not require any order sequence. Note that SAM and EW sites considered 'SAM-type, radiating' targets are already assumed to be targetable by TACOPS. These units do not have to be assigned target numbers. The model will automatically assign the site i.d. number (from the data block) as their target number. Only the other remaining IADS elements that will be a target need to be specified. In this scenario, EW Reporting Station #1 is non-SAM target #1. This is specified by the data set {1,2,1} and is found in the sample data file. All

the other remaining four IADS elements are specified as non-SAM targets. The IADS part of this scenario is now completely specified.

2. The Physical Layout

The relative positions of the IADS elements in Figure 5 are not to be interpreted as a physical layout of the IADS and the SAM units. Figure 5 only shows the possible message paths of the IADS network. Before the user creates the input data file, the physical location of all SAM and non-SAM units in the scenario along with all attacker flightpaths must be defined. The physical location of the EW and SAM sites are specified in the SAM data block section of the input data file. The model requires the EW and SAM site's location be specified. Computations of the intercept of the site's envelope with the attackers flightpath require this information. The model does not require position information for the other IADS elements. Although these IADS elements may be assumed to be at the final impact position of the attackers weapons, only its 'non-SAM' target i.d. number is needed to compute the new attrition value. The user might envision the positions of all the elements in the scenario so that the arrangement of targets and SAM units is logical and meaningful.

Figure 6 shows the positional layout of the sample scenario. The EW and SAM sites location are shown. The detection and launch ranges of the EW and SAM sites are also indicated. Figure 7 shows the assumed relative location of the IADS elements not shown in Figure 6. The SAM units are located around the ADWOC giving it maximum protection. The location of the EW Reporting Stations and

Battalion Centers are such that the SAMs give them some protection. This would be a logical arrangement for the SAM units.

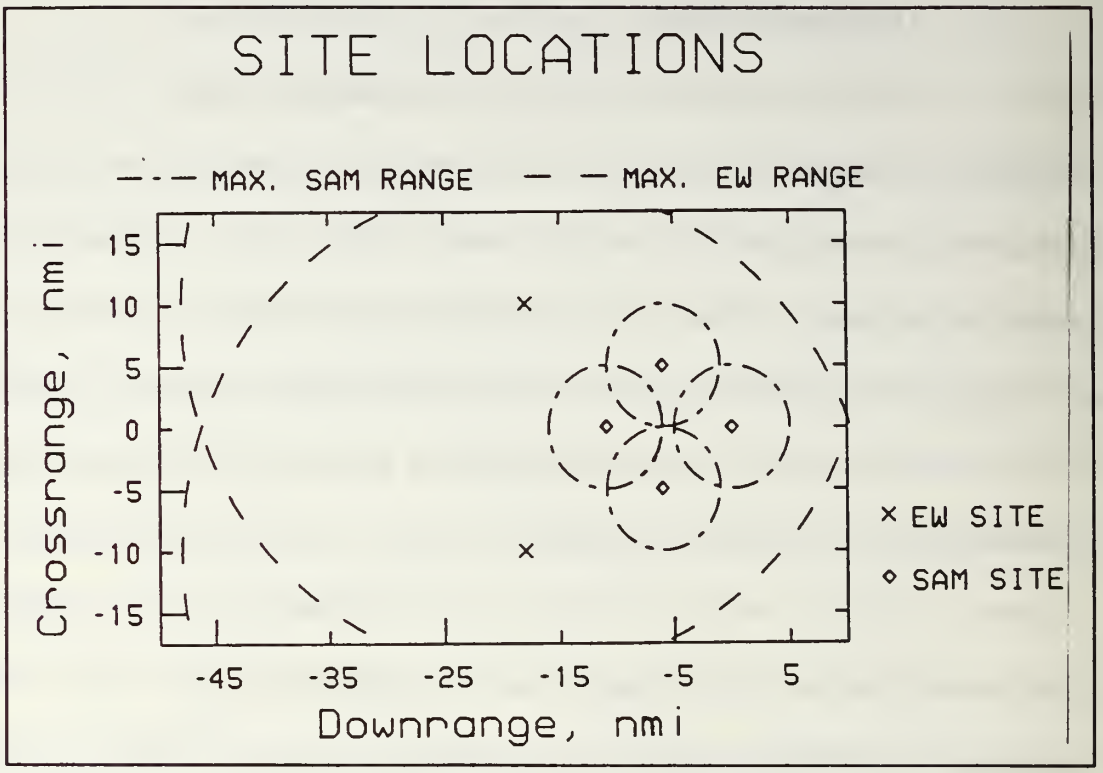

Figure 6 - EW and SAM Site Locations and Max Range

Six attackers will fire a missile 1.0 mile from their targets then turn and exit the area. One attacker is assigned to each site. Two of the attackers will always approach directly from the left, shoot their missiles at the EW sites and turn 180 degrees to the left and exit the area. The four remaining attackers all start at a location 65 miles away from the center of the group of all SAM sites and at a specified approach angle. They fan out from this starting point, approach their targeted sites directly and then turn to the left exiting the area as the first two attackers did. The approach angle is varied to test the sensitivity of the results to the

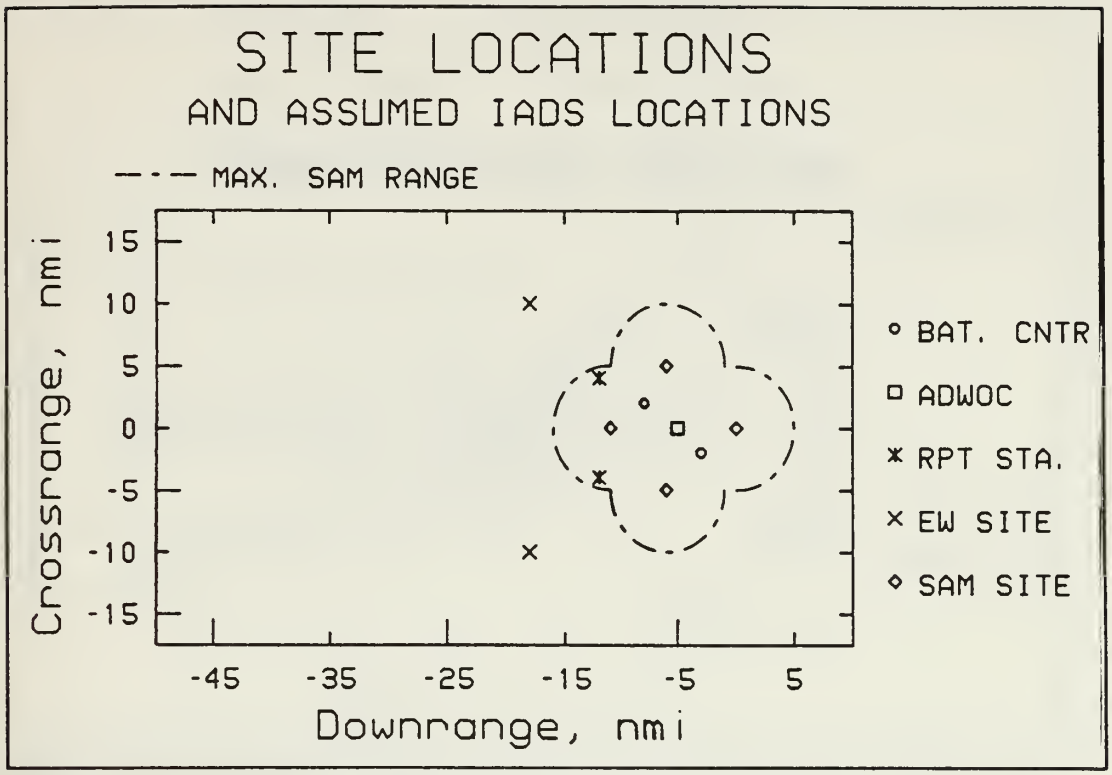

Figure 7 - Site and IADS Elements Assumed Locations

geometry of the scenario. See Figure 8 to understand how the approach angle is defined. The first approach angle starts at 45 degrees clockwise up from a horizontal line. The simulation is run and at attrition values computed and recorded for this approach angle. The simulation is run again using an input data file for a different approach angle. The next approach angle is incremented down (counter-clockwise) from the previous angle by 10 degrees. The angle is incremented until the final angle is 45 degrees below the horizontal line. This gives 10 different approaches. Figure 9 shows the scenario with one such approach angle (-25 degrees). The arrows indicate the direction of the attackers flightpath.

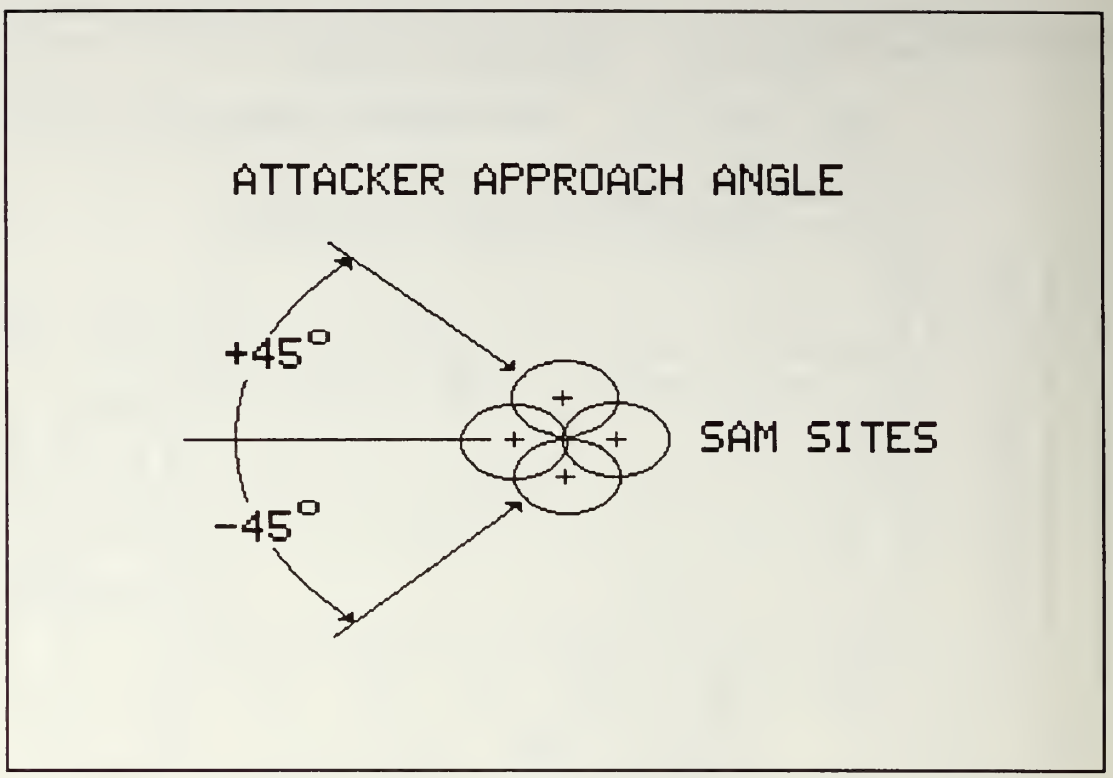

Figure 8 - Attacker Approach Angle Defined

3. Picking the EW Site Delay Time

The EW site time delay has a direct effect on the attackers attrition. This delay is the time that must pass before the EW site sends its detection information to the ADWOC. If the delay is short enough, the detection information will arrive at the ADWOC and then be available to the SAMs before the attackers enter any of the SAM launch envelopes. The number of shot opportunities at the attackers (and the resulting attacker attrition) will be the same as the original TACOPS model. If the EW site delay is so long that the detection information arrives after the last attacker exits all the SAM envelopes, no shot opportunities will be possible.

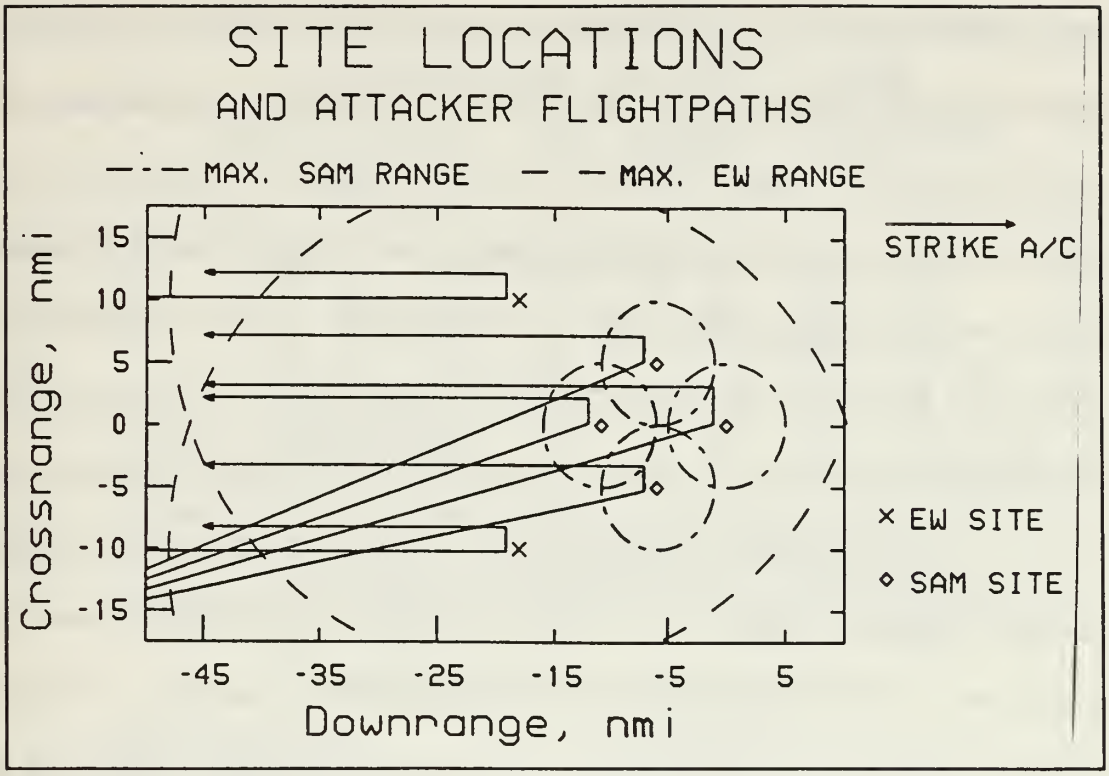

Figure 9 - Attacker Flightpath for -25 Degree Approach Angle

A time delay which resulted in an effect on the attackers attrition was desired. Realistic IADS delay times were not determined. The delay picked was based upon the timing of the attackers in the scenario. The exact time that the first attacker hit its SAM (not EW) site was determined from all the approach angles. Assuming that the SAMs were not previously alerted to any hostile activity, they should not be radiating or trying to engage any targets. Only after an initial hit on one of their SAM sites, should the sites start to engage the now discovered attackers. Due to the geometry differences of the flightpaths, the different approach angles resulted in different times of impact of the first weapon. The earliest weapon impact time of all the different approach angles was determined using the simulation. A time

delay which would result in the detection information arriving at the ADWOC 30 second after this weapon impact was chosen. A time delay was picked and used for all other approach angle scenarios. Using this fixed time delay with the other approach angles results in the detection information arriving, at the latest, 60 seconds after the first weapon impact. That would be the time when the SAMs would be allowed to engage the attackers. This delay should reduce the shot opportunities somewhat but still allow variations in attrition due to the different approach angles.

For this scenario, unclassified and somewhat arbitrary values were picked. The EW maximum detection range was set at 30 miles, the SAM maximum launch range was set at 5 miles and the attackers speed was a constant .125 miles per second. The resulting EW detection of the attackers always occurred at t+125 seconds into the simulation. A EW detection normal processing time delay of 355 seconds met the conditions previously described. This value was used for all the different approach angles.

D. RESULTS

Attacker attrition values were computed for each approach angle using both the original and the IADS TACOPS. The results are summarized in Figure 10. A sample of the output of TACOPS is listed in APPENDIX D. The attrition value used in this section is located at 'LABEL A' in that appendix.

The original TACOPS results are commented on first. The attrition is symmetrical having a plateau for the smaller approach angles and a reduction at both

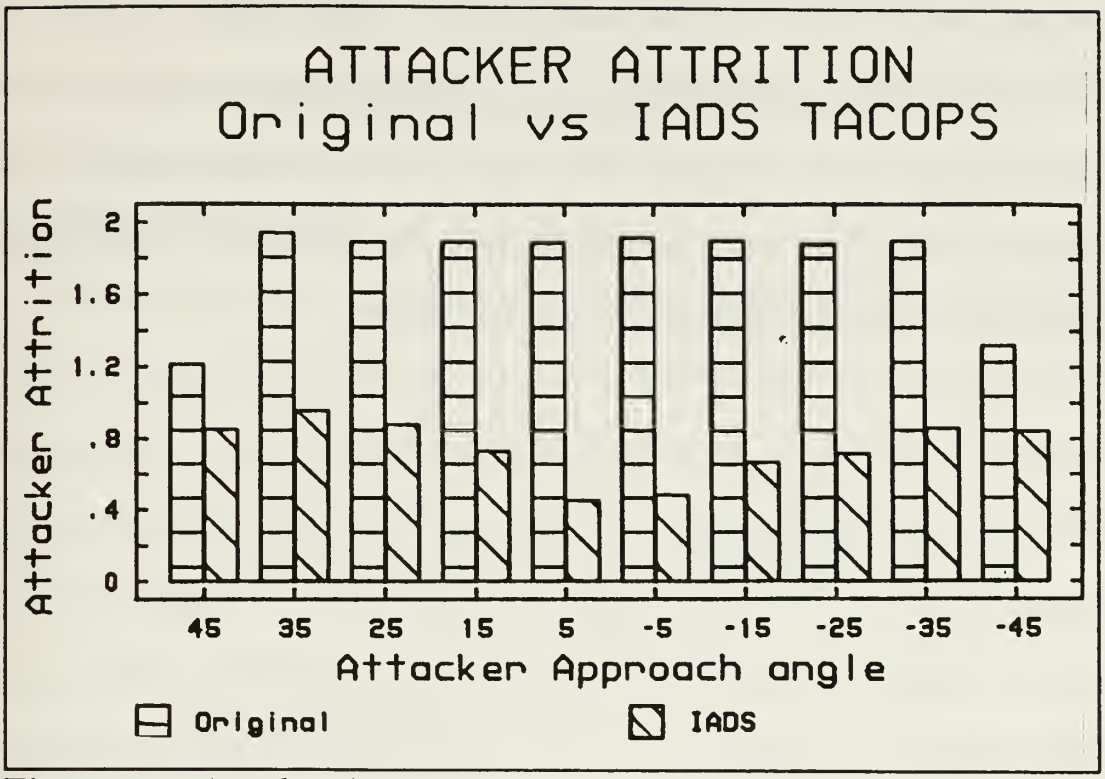

Figure 10 - Attacker Attrition versus Approach Path using Original and IADS TACOPS

extreme angles. Symmetrical attrition values result from the symmetrical relative placement of the SAM sites. A reduction in attacker attrition at the two extreme approach angles is a result of the attackers flightpath. At these two extremes, the attackers flightpath will yield less shot opportunities than the other angles.

The IADS TACOPS values are all below the original TACOPS results. The selected IADS delay significantly reduces the shot opportunities. This results in a reduction in attacker attrition compared to the original model. The variations of attacker attrition versus approach angle are the result of attacker flightpath differences (directly due to angle-of-approach differences) when assuming the fixed IADS delay. The major variation in attrition is the depression for small approach

angles. Relative to when the first weapon impact occurs, a fixed time delay lets the SAMs start engaging the attackers at a different time for each approach angle. Small approach angles have the longest delay after first weapon impact compared to larger angles. This results in less shot opportunities and subsequently less attrition. This explains the central depression for the IADS results.

VII. PROGRAM DOCUMENTATION

A. IADS SUBROUTINES

Many new subroutines were added to TACOPS to incorporate the IADS model capability. Figure 10 shows all the new subroutines developed for the IADS model. Note that some of the original TACOPS subroutines were slightly modified to allow the IADS functions to reside in the program. These are indicated in this figure with a darkened triangle in the upper right corner of the subroutine block. Complete listings of these modified subroutines are included with the listings of the IADS subroutines in APPENDIX D. Also included in this appendix is a listing of some subroutines developed and used in the testing of the IADS program. They may be useful when expanding and debugging the program.

B. DOCUMENTATION IN THE IADS SUBROUTINES

At the beginning of each subroutine is a block of comment lines. These comments include 1) an overall description of the purpose of the subroutine and how the subroutine will accomplish the purpose, 2) the name of the calling subroutine, and 3) the names of all subroutines called by this subroutine. Within the code, comment lines are placed to help the user to understand the meaning of the code.

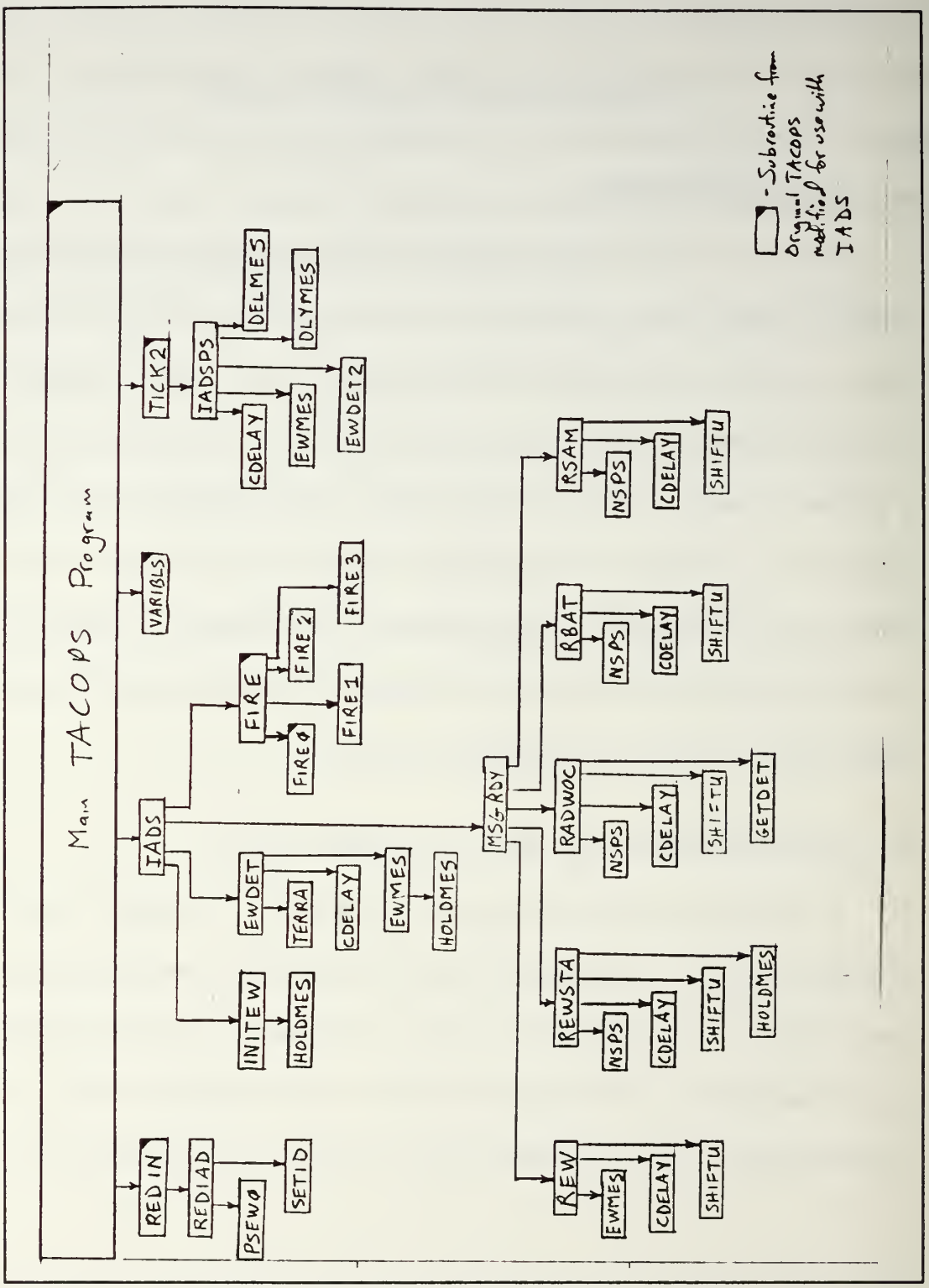

Figure 11 - Block Diagram of TACOPS and the new IADS Subroutines

C. MEANING OF THE 'INCLUDE' STATEMENT

A FORTRAN statement is used in the IADS subroutines that is available at the Naval Postgraduate Schools mainframe computer that may not be universally available. That statement is the INCLUDE statement. The format of this statement is "INCLUDE 'filename filetype'". This directs the statement to locate and use the lines contained in the designated file. When the code is compiled, the lines in the file are included into the code as part of the subroutine.

This statement made using named COMMON statements easier during the program development time. It allowed changing a named COMMON block that many subroutines used by changing the contents of only one file. Without this statement, all the subroutines using the COMMON block would have to be changed separately. Not only would this take more work but there would be the chance that the modifications would not be identical (which is required). All files used with the INCLUDE statement were given the filetype 'DEF'. They are listed after the code in APPENDIX D.

D. CREATING THE INPUT DATA FILE

As noted in the chapter on the sample scenario, creating the input data file before TACOPS can run is demanding. There exists little documentation on the inputs, their bounds and implications, their format and required position in the data file.

APPENDIX B contains a sample input data file that has been annotated to help the user understand program input. The annotations focus on the format, implications and required position of the input values in the context of the whole file. Examples of different model options and the resulting required input needs are shown. The annotations also identify the variable names the input values will be associated with in the code.

E. RUNNING TACOPS WITH AND WITHOUT THE IADS

While TACOPS has been expanded to include an IADS, it is still possible to run the model in its original modes. The original fire control logic modes were selected by inputting a specific number for the third value in the first data line in the input data file. Originally, only the values from 0-1 had any meaning. These were expanded to have meaning for the values of 0-5. Only one IADS mode (a value of 5) was developed so the IADS model could be exercised. Developing better target allocation algorithms will expand these choices further. This following list is the value and the resulting fire control logic that TACOPS will use.

1. 0 = Free-fire doctrine, non-IADS mode; if more that one target is engageable by the same site, select the one with the highest Ps and then the shortest SAM flight time.

2. 1 = Maximum of one SAM unit engaging one attacker at a time (one-on-one), non-IADS mode; if more that one target is engageable by the same site, select the one with the highest Ps and then the shortest SAM flight time.

3. 2 = Maximum of two SAM units engaging one attacker at a time, non-IADS mode; if more that one target is engageable by the same site, select the one with the highest Ps and then the shortest SAM flight time.

4. 3 = Similar to the free-fire doctrine (non-IADS mode), but if more that one target is engageable by the same site, selection is not made using the highest Ps or the shortest SAM flight time. The first attacker found to be engageable while going through the loops checking all attackers will be selected.

5. 4 = Similar to the one-on-one doctrine, but if more that one target is engageable by the same site, selection is not made using the highest Ps or the shortest SAM flight time. The first attacker found to be engageable while going through the loops checking all attackers will be selected.

6. 5 = Identical to value 0 (free-fire) but using IADS mode.

When an IADS mode simulation is desired, the input data file will contain all the originally needed data plus the IADS data. The user can run the same data file with all the different fire control logic options just by changing the previously described fire control logic input value. Thus with an IADS data file, non-IADS modes still work. The reverse situation is obviously not allowed as the IADS data would not be specified.

F. DEBUG OPTIONS

The original TACOPS has print statements imbedded into the code and are selected by a user input value when TACOPS first starts running. The times of significant events and results of important computations are printed to a file for future study. This is called the 'DEBUG' level. The higher the number, the higher the level of detail. The subroutines developed for the IADS model continued this procedure. The user is cautioned about selecting the 'DEBUG' level. A level of 1 will give the most significant events of the simulation. A level of 2 will show some of the IADS message traffic and some TACOPS intermediate calculations. This will also

easily be 10 times the size of output from level 1. A level of 3 gives output for all the imbedded printout statements and may fill up the users disk space before the simulation is done.

VIII. MOEs OF THE IADS

This model may be run by simulating different combinations of weapons and tactics to explore the different results of the systems and tactics. The conclusions reached when comparing these combinations against each other will be greatly influenced by the choice of MOEs. A variety of MOEs are described in this chapter which are indicators of the effectiveness of IADS with selected systems and tactics against a threat. The MOEs explored are: 1) attrition against the attackers; 2) attrition of the IADS-defended units; and 3) indicators which show the degradation of the IADS functions. Some subtle implications of these MOEs are also described.

A. ATTRITION AGAINST THE ATTACKERS

1. Maximum Total Damage to the Attackers

The TACOPS model computes the expected number of attackers that survive at the conclusion of the simulation. The total attrition of the attackers is computed by subtracting the number of attackers that survive from the initial number of attackers. The total attrition against the attackers relates the effectiveness of the IADS model and can be used as an MOE. Since the IADS' purpose is to defend its units, it would do this by inflicting the maximum damage (attrition) on the attackers. With this attained, it could be supposed that the attackers effectiveness would be at a minimum. Thus any damage inflicted by the attackers on its targets would assumed to be at a minimum.

When the importance value of each IADS unit is the same and the threat from each attackers is also the same, maximizing the total damage to all attackers might be a good approach to minimizing the damage to the IADS and a good MOE. If each targets' value or attackers' threat is not the same, this approach might allow unacceptable damage to occur to specific high value targets while minimizing the total damage to all targets. To correct this situation, the value of the target should be included in the MOE.

B. ATTRITION OF THE IADS-DEFENDED TARGETS

The model computes the expected number of IADS-defended targets remaining. Similarly to the computation of the attackers attrition, this number is used to compute the total attrition of the attackers targets. This total attrition could be an MOE that indicates the attackers effectiveness in damaging its targets. Similarly to the case of the attackers attrition, when all targets are of equal value the sum of the attrition of these targets may a valid MOE. When the values are not equal, the sum may have to be weighted to account for the difference in value. Another MOE might be to focus on the attrition to the high value target alone.

C. INTERRUPTION OF IADS RELATED FUNCTIONS

Reducing the effectiveness of the IADS may be the objective when running this model. There would a variety of MOEs which would be legitimate indicators. A couple of them are presented.

1. Number of SAMs launched

A final output of the model is the total number of missiles each SAM site has remaining available to launch after the simulation has ended. This indicates a residual fighting capability that was not used during the simulation. A large number of remaining missiles would indicate that the IADS was not functioning as well as it could, especially if this correlated with a reduction in the total attrition to the attackers.

The implications of this MOE could be dominated by effects in the model. If the same number of SAMs were used but the allocation of them was different, the new attrition might show a significant difference (which could be higher or lower attrition). Drawing the correct conclusions from this situation would be complex. It must be remembered that the allocation algorithm can have a major influence on the total attrition of attackers; thus the residual fighting capability alone is not a strong MOE.

2. Delay in Transferring Messages

The timely transfer of messages is crucial to the functioning of the IADS. Damaging the IADS elements will result in extra delay. As shown in the sample scenario, with sufficient damage, the IADS will not function effectively. Destroying parts of the IADS will cause the messages to arrive by alternative paths. These paths will usually involve more delay than the primary path. Although not a present output of the model, the user could modify the code to output the delays.

It is not appropriate to use this MOE alone. To be useful, this MOE must be correlated with one measuring attrition. Until the increased delays start to influence the engagement opportunities of the SAM units, the model will not show any change in attrition values. Beyond a certain delay, no more engagements are possible and any further delays will not be apparent. The results will typically look for delays which have significant impact on attrition. Correlation of the delays and attrition will identify the influential delays.

D. CONCLUSION

Simulations are used to produce the data in order to answer specific questions. This model can produce a large variety of information relating to the intermediate and final results of an IADS. The appropriate choice of MOEs will help to answer the original questions. The inappropriate choice of MOEs will, at best, confuse the issue and, at worst, imply incorrect answers.

IX. CONCLUSIONS

A. TACOPS DOCUMENTATION

Documentation of the TACOPS program was virtually nonexistent. This thesis presents an overview of the program, a description of the type of model used in the simulation and a detailed description of the detection, engagement and fire control processes modeled in TACOPS. The assumptions and deficiencies in these areas were highlighted.

B. THE INTEGRATED AIR DEFENSE SYSTEM

Integrated air defense systems exist throughout the world. They are diverse in function and capability. The overall functions and structure of an IADS and the individual components were presented. Ways of degrading the performance of the IADS were also presented. Time delay in information transfer was consistent with the modeling assumptions of TACOPS and used as the appropriate measure of IADS performance. Its effect on the IADS performance and some methods to influence it were described.

C. IADS ALGORITHMS

The major functions of an IADS were written as computer algorithms and incorporated into TACOPS. These include EW detection of the attackers, message generation and transfer, data fusion nodes for message traffic control and decision

nodes where the data in collected and acted upon. These IADS elements may be damaged by the attackers and the IADS functions degraded.

The expanded TACOPS model can be run in its original mode or with the IADS functions selected. Documentation of the IADS algorithms is provided in the form of; 1) an operating description; 2) a sample scenario; 3) a sample input data file; 4) a sample output data file; 5) a complete code listing.

Some IADS functions were not modeled. One of them was the effects of jamming the EW sites. TACOPS models the effects of jamming of the SAM sites. Since the modeling of EW and SAM sites is essentially identical, the jamming subroutine in TACOPS could be easily adapted to simulate the effects of EW jamming. It is left for a future user to expand this area.

Target allocation was another IADS functions not modeled. Optimization in this area would be a thesis subject unto itself. Once developed, its algorithms could be incorporated with the ones presented here.

D. SUMMARY

The inclusion of these IADS algorithms has improved the realism of TACOPS. In addition, TACOPS can now be used to evaluate tactics against the IADS elements themselves.

The author has presented much about the functions, components and structure of an IADS. The benefits of a directed graph approach to simulate information transfer have been reinforced.

The advantages of using the appropriate model and programming language for a simulation have also been reinforced. TACOPS is a hybrid of a discrete event and time step model written in the FORTRAN77 programming language. A considerable amount of work was needed to develop the model. As the model was written, modifications in one area required modifications in many other areas of the model. Expanding the model further is expected to take a good deal of effort. A purely discrete event step model and an object orientated programming language would have required less effort develop and should be easier to expand.

LIST OF REFERENCES

1. Bernota, Thomas J., Lt., USNR Masters Thesis, A Model for the Evaluation of C^3CM Effects On an Integrated Air Defense System, September 1990

2. Fowler, Charles A. and Nesbit, Robert F., "Tactical C^3, Counter C^3, and C^5", Journal of Electrnic Defense", November/December 1980

3. Fitts, Richard E., LtCol., USAF "The Strategy of Electromagnetic Conflict", Peninsula Publishing, Los Altos, California, 1980

4. Scheler, D. Curtis, Ph. D., Introduction to Electronic Warfare, Artech House Inc., Norwood, Massachusetts, 1986

5. Atchision, Richard M., Col., USAF " Electronic Combat, Threat, Doctrine, Technology, "Journal of Electronic Defense", April 1987

BIBLIOGRAPHY

Hansen, James, "The Development of Soviet Tactical Air Defense", International Defense Review, May 1981

Przemieniecki, J. S., Air Force Institute of Technology, "Introduction to Mathematical Methods in Defense Analyses", AIAA Education Series, 1990

APPENDIX A

A Form for Evaluating

Each IADS Directed Graph Connections

BLANK FORM

IADS Type to be Studied:_____

What is the required response from this type when it:

Receives a Message from EW Site:

Does it send a message to:

EW Site:

EW Reporting Station:

ADWOC:

Battalion Center:

SAM:

Receives a Message from an EW Reporting Station:

Does it send a message to:

EW Site:

EW Reporting Station:

ADWOC:

Battalion Center:

SAM:

Receives a Message from the ADWOC:

Does it send a message to:

EW Site:

EW Reporting Station:

ADWOC:

Battalion Center:

SAM:

Receives a Message from a Battalion Center:

Does it send a message to:

EW Site:

EW Reporting Station:

ADWOC:

Battalion Center:

SAM:

Receives a Message from a SAM site:

Does it send a message to:

EW Site:

EW Reporting Station:

ADWOC:

Battalion Center:

SAM:

EXAMPLE OF FORM FILLED IN

IADS Type to be Studied: EW Reporting Station

What is the required response from this type when it:

Receives a Message from EW Site:

(This type normally passes messages along and makes copies for the multiple connections. If the ADWOC is dead and an EW site sends a message, this type will generate an 'alert' message)

Does it send a message to?:

EW Site: No connection.

EW Reporting Station: Pass copy to other Stations

ADWOC: Pass copy to ADWOC

Battalion Center: Send only if an 'alert'

SAM: Send only if an 'alert'

Receives a Message from an EW Reporting Station:

Messages from another EW Reporting Station will be routed depending upon their source and final destination. If the message is from an EW site, its final destination will be the ADWOC. If the message is from the ADWOC, its final destination will be an EW site. If the message is an 'alert', copies are passed to this Stations Battalion Centers and SAM units.

Does it send a message to:

EW Site: Send only if this is the final destination

EW Reporting Station: Send copy of message

ADWOC: Send only if this is the final destination

Battalion Center: Send only if an 'alert'

SAM: Send only if an 'alert'

Receives a Message from the ADWOC:

A message from the ADWOC will be directed to the EW sites. Pass messages based upon the final destination (specific EW site #). Pass a copy to other EW Reporting Stations.

Does it send a message to:

EW Site: Send only if this is the final destination

EW Reporting Station: Send copy of message

ADWOC: Do not send copy of message

Battalion Center: Do not send copy of message

SAM: Do not send copy of message

Receives a Message from a Battalion Center:

The sample scenario assumes that there are no messages from any Battalion Centers to any EW Reporting Stations.

Does it send a message to:

EW Site: No connection

EW Reporting Station: No connection

ADWOC: No connection

Battalion Center: No connection

SAM: No connection

Receives a Message from a SAM site:

The sample scenario assumes that there are no messages from any SAM units to any EW Reporting Stations.

Does it send a message to:

EW Site: No connection

EW Reporting Station: No connection

ADWOC: No connection

Battalion Center: No connection

SAM: No connection

The resulting IADS connection paths are represented in the following figures. The paths are separated into messages coming from and going to the EW Reporting Station. Station #1 is used in this example.

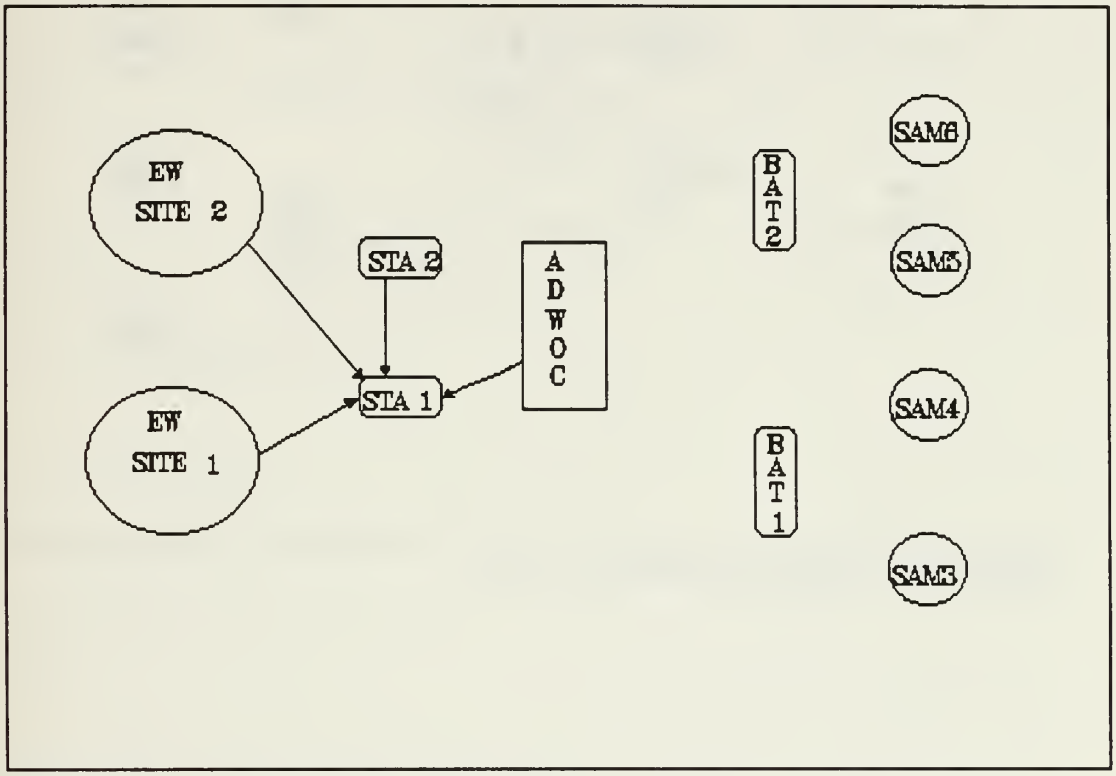

Figure 12 - Incoming Message Paths

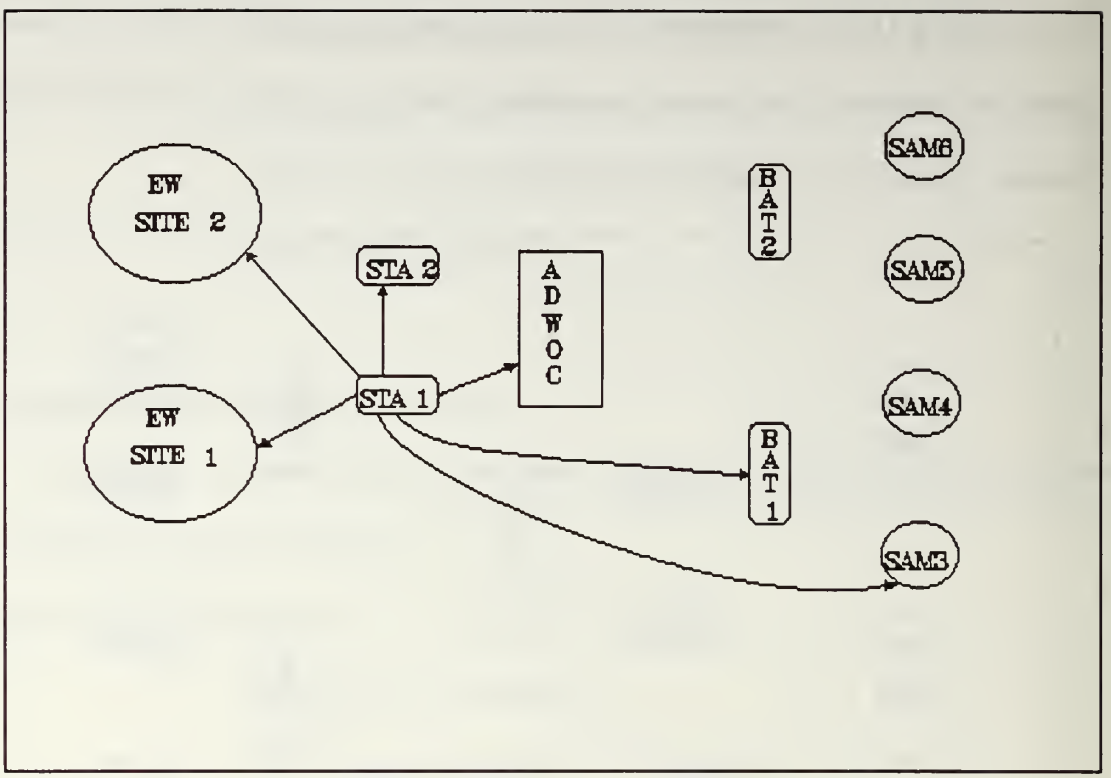

Figure 13 - Outgoing Message Paths

APPENDIX B

An Annotated

Input Data FIle

.....7/17/91..BRYAN F. SMITH WEAPON PLANNING GROUP, CODE 304
.....NAVAL WEAPONS CENTER, CHINA LAKE, CA 93555

.....UNCLASSIFIED.....

INCLUDES UP TO MOD 11

THIS IS A DOCUMENTATION OF THE STANDARD TACOPS DATAFILE
TEXT PRECEEDED WITH A ARE COMMENTS ABOUT THE DATA. THOSE LINES
WITHOUT A ARE THE ACTUAL DATA.

THIS DOCUMENTATION COVERS THE FOLLOWING MODS:

1) NTARGETS ADDED TO HAVE ATTRITION ON NON-SAM TYPE TARGETS
2) NPENTYPE,PENTYPE,MDEST(I,2) ADDED FOR MORE THAN ONE TYPE OF STANDOFF WEAPON (i.e. SLAM/AIWS/WALLEYE) TO BE USED. PENTARPK(*) ALLOWS EACH PENETRATOR TYPE TO HAVE A UNIQUE PK AGAINST THE DIFFERENT SAM TYPES AND THE NON-SAM TARGETS
3) ZS(J) ADDED FOR EACH SAM TO HAVE ITS OWN ALTITUDE.
4) THE VARIABLE PKM WAS CHANGED TO THE ARRAY PKM(9) TO ALLOW THE ARM TO HAVE A UNIQUE PK AGAINST EACH OF THE 9 SAM TYPES.
5) BOM(I),BOMPT(I),BOMTGT(I,1),BOMTGT(I,2),LAUNCH(I),LNCHPT(I), LNCHWPN(I) ADDED TO ALLOW A/C TO DROP BOMBS AND LAUNCH PENETRATORS IF THEY SURVIVE TO THE RELEASE POINT.
6) CHANGED NUMSHT(9) TO NUMSHT(JM) AND ADDED THIS AND PS(J) TO THE SITE DATA TO HAVE EACH SITE WITH A DEFINED STARTING PS AND NUMBER OF AVAILABLE SALVOS.
7) CORRECTION TO PARM FLIGHT PATH COMPUTATIONS; SUB FUTLOK REPLACES SUB LOCKON - WILL NOT ALLOW SITES TO FIRE IF, AT INTERCEPT, THE FUZE ANGLE WILL BE BAD; EXPANDED DOCUMENTING OUTPUT DATA FOR ZTR PROGRAM IN THE AREAS OF SAM SITE RADIATING STATUS AND ARM FUNCTIONS; EXPANDED MODEL TO ALLOW AIRCRAFT TO LAUNCH AND CONTROL MULTIPLE TYPE 1 AND TYPE 2 STANDOFF WEAPONS; DEFINED TYPE 1 STANDOFF WEAPON TO BE A FIRE AND FORGET TYPE, AND A NEW TYPE 2 STANDOFF WEAPON THAT REQUIRES POST-LAUNCH CONTROL OR DESIGNATION;EXPANDED CODE TO ALLOW EACH A/C TO HAVE UP TO 4 BOMB DROP POINTS, LAUNCH UP TO 4 TYPE 1 AND/OR TYPE 2 WEAPONS, AND CONTROL UP TO 4 TYPE 2 WEAPONS
8) CORRECTED CODE IN SHOT SUBROUTINES TO NOT ALLOW SITES TO FIRE AND INTERCEPT PENETRATORS THAT ARE OUT OF ENVELOPE OR ARE TERRAIN MASKED; ADDED TO INPUT DATAFILE VALUES FOR THE MAXIMUM WAIT TIME BEFORE THE SITE LOSES THE TRACK FOR TERRAIN MASKING AND THE OUT OF ENVELOPE CONDITION WHILE A SAM IS ENROUTE
9) ALLOWS 1-1 AND FREE-FIRE SAM FIRING DOCTRINE. IOP(3)=0 SPECIFIES FREE-FIRE (PREVIOUS MODE), IOP(3)=1 SELECTS FIRING DOCTRINE THAT WILL ONLY ALLOW A MAXIMUM OF ONE SAM TO ENGAGE A TARGET.

ALSO IF IOP(4)=1, THIS MAKES A LAUNCH AIRCRAFT NOT LAUNCH ITS POST-LAUNCH CONTROLLED TYPE WEAPON IF THE ASSOCIATED CONTROLLING AIRCRFT HAS ALREADY BEEN DESTROYED AT TIME OF LAUNCH. IF IOP(4)=0 THE WEAPON IS LAUNCHED REGARDLESS OF THE STATE OF THE CONTROLLING AIRCRAFT (THE PREVIOUS MODE) AND THE PK OF THE WEAPON AGAINST THE TARGET IS 0.0 IF THE CONTROLLING AIRCRAFT HAS BEEN DESTROYED. THIS INPUT DATAFILE IS COMPATIBLE WITH THE PREVIOUS MODE(8).

10) ALLOWS SAM SITES TO BE GROUPED TOGETHER TO SIMULATE A MULTI-TARGETTING SAM SYSTEM. DATAFILE NOT COMPATIBLE WITH PREVIOUS MODES.

11) ALLOWS IADS TO BE MODELED. NOT COMPATIBLE WITH PREVIOUS MODES.

TITLE

THE FIRST LINE IS THE TITLE AND CAN BE 70 CHARACTERS LONG.
IT SHOULD CONTAIN THE DATE AND A BRIEF DESCRIPTION OF THE RUN SO
AS TO HELP THE PROGRAMMER UNDERSTAND ITS UNIQUE ATTRIBUTES.

2/1/88 MOD 8 FORMAT UNCLASSIFIED DATA VALUESè

IOP(*) ARRAY DETERMINES THE DOCTRINE THE SAM SITES WILL RADIATE WITH
AND WHICH MODELING (EXPECTED VALUE OR STOCHASTIC) WILL BE USED
IOP(1)=0 - ATTRITION MODELLING: EACH TIME AN A/C IS INTERCEPTED BY A
- SAM SALVO, THE NEW A/C PS = OLD PS - PK(SALVO)*PS(SITE).
- WHEN THE A/C PS GETS BELOW THE MIN PS THRESHOLD, THE A/C
- IS 'KILLED' AND REMOVED FROM THE SIMULATION.
IOP(1)=1 - STOCHASTIC MODELING: EACH TIME AN A/C IS INTERCEPTED BY A
- SAM SALVO, THE A/C PS = 1/0 DEPENDING UPON A RANDOM # DRAWN.
- IF A/C PS=0, A/C IS 'KILLED' AND REMOVED FROM THE SIMULATION
IOP(2)=0 - REGULAR SITE RF EMMISION DOCTRINE
IOP(2)=1 - PERFECT(ONLY AFTER LAUNCH) SITE RF EMMISION DOCTRINE
IOP(3)=0 - FREE-FIRE (REGULAR) SITE LAUNCH DOCTRINE
IOP(3)=1 - MAX OF 1-1 SITE FIRING DOCTRINE
IOP(3)=2 - MAX OF 2-1 SITE FIRING DOCTRINE
IOP(3)=3 - FREE-FIRE DOCTRINE W/O MAX PS AND MIN TF SELECTION TESTS
IOP(3)=4 - MAX OF 1-1 DOCTRINE W/O MAX PS AND MIN TF SELECTION TESTS
IOP(4)=0 - REGULAR AIRCRAFT DOCTRINE FOR LAUNCHING TYPE 2 WEAPONS
- I.E. DO NOT CONSIDER IF CONTROLLING AIRCRAFT IS DEAD
IOP(4)=1 - NEW AIRCRAFT DOCTRINE FOR LAUNCHING TYPE 2 WEAPONS
- I.E. DO NOT LAUNCH WEAPON IF CONTROLLING AIRCRAFT IS DEAD
IOP(10) - # OF TIMES TO RUN THE DATA
IOP(3)= (0,1,2,3,4) NON-IADS MODE
IOP(3) = (5,...) IADS MODE

0,0,0,0,0,0,0,0,0,1

NAC,NSITE,PSAMIN,NPENTYPE,NTARGETS

NAC - MAX # OF AIRCRAFT(PENETRATORS) USED FOR THIS DATAFILE.
- VALUE OF PARAMETER IM IN FORTRAN CODE SETS THE UPPER LIMIT.
- PRESENTLY SET TO 30
NSITE - MAX # OF SAM AND EW SITES IN THE DATAFILE.
- VALUE OF PARAMETER JM IN FORTRAN CODE SETS THE UPPER LIMIT.
- PRESENTLY SET TO 60
PSAMIN - MINIMUM SURVIVAL PROBABILITY FOR THE AIRCRAFT BEFORE THE
- PENETRATOR IS DESTROYED AND THUS TAKEN OUT OF THE GAME.
NPENTYPE - # INDICATING THE HIGHEST STANDOFF WEAPON TYPE USED
NTARGETS - MAX # OF NON-SAM AND IADS TARGETS ASSUMED FOR THIS RUN

4,3,0.4,2,0

MAXMS - MAX # OF MULTI-TARGETTING SAM GROUPS IN DATAFILE
- IF = 0, NEXT DATA IS SAM SITE LOCATION AND WEAPON PARAMETERS
- IF = 1, READ IN J=1,MAXMS MSID(J,1) AND MSID(J,2)
- J = 15 PRESENT MAXIMUM
MSID(J,1) - LOWER SAM I.D. # WHICH IDENTIFIES THE START OF THE GROUP
MSID(J,2) - UPPER SAM I.D. # WHICH IDENTIFIES THE END OF THE GROUP

XS(J),YS(J),ZS(J),VM(J),GDT(J),PPK(J),MSITE(J),THETJ(J),CKON1(J),
GTFUS(J),PS(J),NUMSHT(J)

IF IN A NON-IADS MODE, (I.E. IOP(3)=(0,1,2,3,4))
EACH LINE HOLD THE ABOVE DATA FOR ONE SAM SITE. THE FIRST LINE IS FOR SAM SITE #1, THE SECOND LINE WILL BE FOR SITE #2,ETC...(J=1-LAST SITE)

IF IN A IADS MODE, (I.E.IOP(3) = (5,...))
EACH LINE HOLD THE ABOVE DATA FOR ONE SAM SITE AS DEFINED FOR THE NON-IADS MODE. BUT SOME OF THE SAM SITES WILL BE REDESIGNATED AS EW SITES LATER ON IN THE DATA FILE. NOT ALL SAM SITE DATA WILL APPLY TO THE EW SITES BUT MUST BE PRESENT IN THE PROPER FORMAT TO BE READIN CORRECTLY.

XS(J) - X POSITION OF SAM SITE J IN NMI FROM SELECTED REFERENCE POINT
YS(J) - Y POSITION OF SAM SITE J IN NMI FROM SELECTED REFERENCE POINT
ZS(J) - Z POSITION OF SAM SITE J IN NMI FROM SELECTED REFERENCE POINT
VM(J) - AVERAGE SAM MISSILE VELOCITY IN NMI/SEC
GDT(J) - KILL ASSES DELAY TIME (IN SECONDS) FOR THIS SAM SITE
PPK(J) - PK OF SAM SALVO
MSITE(J) - TYPE (DATA BLOCK #1-9) OF SAM SYSTEM AT THIS LOCATION
- ALSO USED TO READ IN LAUNCH BOUNDRY DATA LATER ON
THET(J) - MASKED SITE ANGLE (IN RADIANS) DUE TO A JAMMING A/C
CKON1(J) - RADAR BURN-THROUGH RANGE VALUE
GTFUS(J) - MIN. INTERCEPT ANGLE (IN RADIANS) USED TO DETERMINE IF SAM
- SUCSESSFULLY ENGAGED TARGET
PS(J) - INITIAL SURVIVAL PROBABILITY OF THE SITE. USED IN MULTIPLE
RAIDS WHERE DAMAGE FROM THE PREVIOUS RAID MUST BE MAINTAINED.
NUMSHT(J) - THE TOTAL NUMBER OF SALVOS THAT THIS TYPE OF SAM CAN FIRE
- DURING THIS RUN. FOR SINGLE RAIDS, THIS WILL BE THE SITES

- FULLY LOADED VALUE; FOR MULTIPLE RAIDS, THIS WILL BE WHAT
- THE SITE CAN HAVE TIME TO RELOAD TO.

-05.00, 5.00, 0.0, 0.3000, 1, 0.11, 1, 0.1, 1, 0.4000, 1.00, 6
-10.00, 10.00, 0.0, 0.3000, 5, 0.22, 2, 0.1, 1, 0.4000, 0.90, 4
-10.00, -1.00, 0.0, 0.3000, 9, 0.33, 3, 0.1, 1, 0.4000, 0.80, 2

DATA DESCRIBING THE AIRCRAFT AND STANDOFF WEAPON TYPE ARE NOW READ. EACH BLOCK OF DATA DESCRIBE THE PENETRATOR. THE FIRST BLOCK WILL BE FOR PENETRATOR #1, THE SECOND FOR PENETRATOR#2, UNTIL THE LAST BLOCK (NAC) IS READ

PENTYPE(I) (20 CHARACTER COMMENT MAY BE ADDED AFTER THE DATA; NO SPACES)
PENTYPE(I) - INTEGER VALUE WHICH INDICATES THE TYPE OF PENETRATOR THIS
- DATA BLOCK DESCRIBES:(0 = AIRCRAFT); (1 = FIRE AND FORGET)
- ;(2 = CONTROL OR DESIGNATION AFTER LAUNCH)

0 TYPE_1_WEAPON_FROM_AIRCRAFT_2

IF PENTYPE(I) = 0; THE PENETRATOR IS ASSUMED TO BE AN AIRCRAFT THAT CAN DROP BOMBS AND/OR LAUNCH AND/OR CONTROL STANDOFF WEAPONS. IT IS REQUIRED THAT THE AIRCRAFT PENETRATOR # BE LESS THAN THE WEAPON PENETRATOR # IT LAUNCHES (i.e. A/C #3 CAN LAUNCH WEAPON # 12 BUT IT SHOULD NOT LAUNCH WEAPON 1 OR 2). THE FOLLOWING DATA WILL DEFINE IF ANY THESE ACTIVITIES WILL BE PERFORMED BY THIS AIRCRAFT.

BOM(I),LNCH1(I),LNCH2(I),CNTRL2(I)

THIS LINE DETERMINES THE ACTIVITY OF THE TYPE 0 (AIRCRAFT) PENETRATOR

BOM(I) - TOTAL # OF BOMBS THIS A/C DROPS (MAX 4)
LNCH1(I) - TOTAL # OF TYPE 1 STANDOFF WEAPONS THIS A/C LAUNCHS (MAX 4)
LNCH2(I) - TOTAL # OF TYPE 2 STANDOFF WEAPONS THIS A/C LAUNCHS (MAX 4)
CNTRL2(I)- TOTAL # OF TYPE 2 STANDOFF WEAPONS THIS A/C CONTROLS (MAX 4)

4,0,0,0

BOMPT(I,N),BOMTGT(I,1,N),BOMTGT(I,2,N)

IF BOM(I) > = 1;THE AIRCRAFT WILL DROP GROUPS OF BOMBS. 3 MORE VALUES ARE NEEDED FOR EACH GROUP OF BOMB DROPS UP TO N GROUPS.

BOMPT(I,N) - THE CHECKPOINT # WHERE BOMB GROUP N ARE TO BE RELEASED.
BOMTGT(I,1,N) - THE TARGET TYPE. 1=SAM SITE TYPE TARGET; 2=NON-SAM SITE
- TYPE TARGET.
BOMTGT(I,2,N) - THE TARGET LOCATION #.
- IF BOMTGT(I,1)=1; THE VALUE IS THE SAM SITE # (1-NSITE)
- IF BOMTGT(I,1)=2; THE VALUE IS THE NON-SAM SITE TYPE
- TARGET # (1-NTARGETS)

2,1,2
2,1,2
2,1,2
2,1,2

L1CKPT(I,N),L1WPN(I,N)

IF LNCH1(I,N)> =1 THEN LINES CONTAINING TWO VALUES EACH MUST FOLLOW WHICH DESCRIBE THE LAUNCH OF EACH TYPE 1 (FIRE AND FORGET) STANDOFF WEAPON DESIGNATED UP TO N LINES

L1CKPT(I,N) - THE CHECKPOINT # OF THE LAUNCH A/C WHERE THE WEAPON
- IS TO BE LAUNCHED.
L1WPN(I,N) - THE WEAPON PENETRATOR # TO BE LAUNCHED FROM THIS A/C.

2,3

L2CKPT(I,N),L2WPN(I,N)

IF LNCH2(I,N)> =1 THEN LINES CONTAINING TWO VALUES EACH MUST FOLLOW WHICH DESCRIBE THE LAUNCH OF EACH TYPE 2 (LAUNCH AND CONTROL) STANDOFF WEAPON UP TO N LINES

L2CKPT(I,N) - THE CHECKPOINT # OF THE LAUNCH A/C WHERE THE WEAPON
- IS TO BE RELEASED.
L2WPN(I,N) - THE WEAPON PENETRATOR # TO BE LAUNCHED FROM THIS A/C.

2,4

C2START(I,N),C2END(I,N),C2WPN(I,N),C2TIME(I,N)

IF CNTRL2(I,N) > =1 THEN LINES CONTAINING THREE MORE VALUES MUST FOLLOW
WHICH DESCRIBE THE ACTIVITY OF THE CONTROL AIRCRAFT
UP TO N LINES.

C2START(I,N) - THE CHECKPOINT # OF THE CONTROL A/C WHERE CONTROL OF THE
- TYPE 2 WEAPON WILL START
C2END(I,N) - THE CHECKPOINT # OF THE CONTROL A/C WHERE CONTROL OF THE
- TYPE 2 WEAPON WILL END
C2WPN(I,N) - THE WEAPON PENETRATOR # TO BE CONTROLLED FROM THIS A/C.
C2TIME(I,N) - THE MAXIMUM TIME (IN SECONDS) BEFORE IMPACT OF THE WEAPON
- # N WHEN CONTROL OF THE WEAPON CAN BE LOST (i.e. THE
- CONTROL A/C # I DESTROYED) AND THE WEAPON WILL STILL HIT
- THE TARGET

2,3,4,30.0

MDEST(I,1),MDEST(I,2)

IF PENTYPE(I)=1 OR =2; DATA ON THE TARGET FOR THIS WEAPON IS NEEDED.

MDEST(I,1) - AN INTEGER VALUE WHICH INDICATED THE TYPE OF TARGET THAT
- THIS PENETRATOR IS PROGRAMMED TO HIT. 1=SAM SITE TYPE
- TARGET;2=NON-SAM SITE TYPE TARGET.
MDEST(I,2) - AN INTEGER VALUE WHICH HOLD DATA ON THE TARGET.
- IF MDEST(I,1)=1; THE VALUE IS THE SAM SITE # (1-NSITE)
- IF MDEST(I,1)=2; THE VALUE IS THE NON-SAM SITE TYPE
- TARGET # (1-NTARGETS)

1,3

NCKPT(I),T(I,1)

IF PENTYPE = 0, THEN THE MAX # OF CHECKPOINTS AND THE START TIME
MUST NOW BE INPUT.

IF PENTYPE(I) > 0, ONLY THE MAX # OF CHECKPOINTS NEEDS TO BE
INPUTTED. THE PROGRAM WILL CALCULATE THE START TIME
OF THE STANDOFF WEAPON.

NCKPT(I) - THE TOTAL # OF LINES OF DATA (TO FOLLOW THIS LINE) WHICH
- DESCRIBE THE FLIGHTPATH OF THE PENETRATOR. THIS IS USED BY
- THE DATA READING PROGRAM TO DETERMINE HOW MANY LINES TO READ

- FOR EACH PENETRATOR. THE MAXIMUM NUMBER OF CHECKPOINTS IS
- SET IN THE FORTRAN CODE BY THE PARAMETER KM AND IS PRESENTLY
- EQUAL TO 15.

T(I,1) - THE TIME IN SECONDS WHEN THE Ith PENETRATOR COMES INTO THE
- SIMULATION. BEFORE THIS TIME, THIS PENETRATOR DOES NOT EXIST.

3,0

X(I,K),Y(I,K),Z(I,K),V(I,K)

THESE LINES OF THE DATA BLOCK HOLD THE A/C CHECKPOINT FLIGHTPATH DATA

X(I,K) - X POSITION IN NMI OF A/C FOR CHECKPOINT # K
Y(I,K) - Y POSITION IN NMI OF A/C FOR CHECKPOINT # K
Z(I,K) - Z POSITION IN NMI OF A/C FOR CHECKPOINT # K
V(I,K) - A/C VELOCITY BETWEEN THIS AND THE NEXT CHECHPOINT IN NMI/SEC

-35.0, 0.0, 0.500, 0.100
-10.0, 10.0, 0.100, 0.100
-40.0, 9.0, 0.500, 0.100

THE REMAINDER OF THE PENETRATOR CHECKPOINT DATA BLOCKS WILL LOOK LIKE THESE BLOCKS BELOW.

0
0,1,1,1
2,3
2,4
2,3,4,30.0
4,0 A/C #2
-35.0, 9.0, 0.100, 0.200
-15.0, 0.0, 0.200, 0.200
-27.5, -5.0, 0.100, 0.200
-35.0, -5.0, 0.100, 0.200

1 TYPE_1_WEAPON_FROM AC_2
1,3
3
-15.00, 0.00, 0.200, 0.20
-12.50, -0.50, 2.000, 0.05
-10.00, -1.00, 0.000, 0.10

2 TYPE_2_WEAPON_FROM_AC_2
1,3
3
-15.00, 0.00, 0.200, 0.20

-12.50, -0.50, 0.100, 0.05
-10.00, -1.00, 0.000, 0.10

PENTARPK(II,1,L),L=1,9

IF ANY STANDOFF WEAPONS ARE BEING USED IN THIS RUN (i.e. NPENTYPE>0), THE PK OF THESE WEAPONS AGAINST 9 SAM TYPES MUST NOW BE LISTED.

0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9 TYPE 1 PK FOR SAM TYPES

PENTARPK(II,2,L),L=1,NTARGETS

IF NON-SAM TARGETS ARE ASSUMED (i.e. NTARGETS>0), ANOTHER LINE MUST NOW LIST THE PK FOR THE FIRST STANDOFF WEAPON AGAINST THESE TARGETS. THE NUMBER OF PK VALUES MUST EQUAL THAT NUMBER (NTARGETS) OF TARGETS.

IF NPENTYPE'S VALUE (# OF STANDOFF WEAPON TYPES) IS GREATER THAN 1, ANOTHER TWO LINES MUST FOLLOW IN THE SAME FORMAT AS THE PREVIOUS TWO WHICH HOLD THE PK VALUES FOR THE OTHER STANDOFF WEAPON TYPES AGAINST THE TARGETS. IN THIS CASE, ONLY ONE MORE TYPE OF STANDOFF WEAPON IS INDICATED (NPENTYPE = 2) SO ONLY TWO MORE DATA LINES ARE NEEDED.

0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9 TYPE 2 PK FOR SAM TYPES

BOMPK(1,L),L=1-9

IF ANY OF THE A/C ARE TO DROP BOMBS, THE PK OF THE BOMBS AGAINST THE TARGETS MUST NOW BE INPUT. SIMILAR TO THE FORM ABOVE, THE FIRST LINE IS FOR THE BOMBS AGAINST THE 9 SAM TYPES.

0.2,0.1,0.3,0.4,0.5,0.6,0.7,0.8,0.9 BOMB PK FOR SAM TYPES

BOMPK(2,L),L=1-NTARGETS

IF ANY NON-SAM TYPE TARGETS ARE SPECIFIED FOR THIS RUN, THE BOMB PK AGAINST THESE TARGETS MUST NOW BE LISTED EVEN IF THE BOMBS WILL ONLY BE USED AGAINST THE SAM TYPE TARGETS.

ISECT(I,J);I = 1,NAC:J = 1,NSITE

FOR EACH A/C, A LINE MUST EXIST WHICH INDICATES WHETHER EACH SAM SITE LOCATION (#1 - NSITE) CAN(1) OR CANNOT(0) FIRE AT THIS PENETRATOR

1,1,1
1,1,1
1,1,0
1,1,0

NHRMAC,TMEM

DATA ON THE USE OF 'ARM' WEAPONS FOR THIS RUN

NHRMAC - TOTAL NUMBER OF ARM CARRYING A/C.
TMEM - TIME (IN SECONDS) DURING WHICH THE ARM WILL STILL HIT THE SAM
- SITE EVEN THOUGH THAT SITE HAS STOPPED RADIATING.

1,30

PKM(9)
PKM - PK OF ARM AGAINST THE 9 SAM SITE TYPES (i.e. MUST BE 9 #'S)

0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9 ARM PK FOR SAM TYPES

IDHAC,NUMD(IDHAC),NUMB(IDHAC),NUMR(IDHAC)

ARM FIRING TIMES
IF NHRMAC=0 THEN NO ARM A/C ARE ASSUMED AND NO ARM DATA IS NEEDED.
IF NHRMAC>0 THEN DATA ON THE FIRING TIMES OF THE ARMS IS PRESENTED.

IDHAC - PENETRATOR # FOR WHICH THE FOLLOWING ARM DATA APPLIES
NUMD(IDHAC) - # OF DIRECTIVE ARM MISSILES ON A/C IDHAC (AT PRESENT,
- THIS ARM MODE IS NOT USED AND MAY NOT BE A VALID MODE.)
NUMB(IDHAC) - # OF PREPROGRAMMED ARM MISSILES ON A/C IDHAC
NUMR(IDHAC) - # OF REACTIVE ARM MISSILES ON A/C IDHAC

1,0,3,0 ARM DATA

IF NUMD>0 THEN THE FOLLOWING DATA IS FOR THE DIRECTED ARM. IN THIS EXAMPLE NUMD=0 SO NO DIRECTED ARMS WERE ASSUMED TO BE USED.

DRMTG(IDHAC,JJ),TLDRM(IDHAC,JJ)

THE DATA DESCRIBES THE LAUNCH TIME AND THE SITE TYPE
THAT MISSILE HAS BEEN DIRECTED TO FLY TO

DRMTG(IDHAC,JJ) - LAUNCH TIME (IN SECONDS) FOR ARM #JJ OFF OF
- A/C #IDHAC
TLDRM(IDHAC,JJ) - SAM SITE LOCATION THAT ARM WILL FLY TO

IF NUMB>0 THEN THE FOLLOWING DATA IS FOR THE PREPROGRAMMED ARM. IN THIS EXAMPLE NUMB=8 SO 8 PREPROGRAMMED ARMS WERE ASSUMED TO BE USED.

TLB(IDHAC,JJ) - LAUNCH TIME (IN SECONDS) FOR ARM JJ OFF OF A/C IDHAC

PMPREF(IDHAC,JJ) - SAM SITE TYPE THAT ARM WILL FLY TO HIT IF IT IS
- RADIATING DURING THE TIME THE ARM IS SEARCHING.
- SEE DESCRIPTION OF MSITE(*) IN THE FIRST PART OF
- THIS REPORT FOR THE SAM TYPE AND ITS 'SA-' #.

IF NUMR>0 THEN THE FOLLOWING DATA IS FOR THE REACTIVE ARM. IN THIS EXAMPLE NUMR=0 SO NO REACTIVE ARMS WERE ASSUMED TO BE USED. THE PRESENT CODE DOES NOT READ ANY DATA FOR REACTIVE ARMS.

1,1
5,2
10,3

PSSMIN

START READING SAM SITE DATA

PSSMIN - MIN SURVIVAL PROBABILITY THAT THE SAM SITE MUST MAINTAIN
- TO STAY IN THE SIMULATION

0.50 MIN SAM SURVIVIAL PROBABILITY

LL,TRACK(J),MWAITIM(J),OWAITIM(J)

SAM SITE LAUNCH BOUNDRY VALUES

LL - TOTAL # OF LINES THAT FOLLOW WHICH CONTAIN DATA FOR THIS SAM TYPE
- THE PROGRAM USES TO DETERMINE HOW MANY LINES TO READ AFTER THIS
TRACK(J) - THE TIME (IN SECONDS) THAT THIS SAM SYSTEM TYPE WILL TAKE
- FROM THE START OF THE TRACKING FUNCTION BEFORE THE SYSTEM

- WILL BE READY TO FIRE A SALVO AT THE TARGET.
MWAITIM(J) - MAXIMUM WAIT TIME (IN SECONDS) AFTER THE TARGET IS
- TERRAIN MASKED BEFORE THIS SITE TYPE LOSES TRACK AND
- THE OUTBOUND SAM IS LOST
OWAITIM(J) - MAXIMUM WAIT TIME (IN SECONDS) AFTER THE TARGET IS
- OUTSIDE OF THE SITE LAUNCH ENVELOPE BEFORE THIS SITE
- TYPE LOSES TRACK AND THE OUTBOUND SAM IS LOST

6,10,5,10 SAM TYPE 1

LDUMMY,ESITE1(J,L),ESITE2(J,L)

THIS IS THE RANGE AND ALTITUDE DATA PAIRS THAT OUTLINE THE LAUNCH BOUNDRY FOR THIS TYPE OR SAM SYSTEM

LDUMMY - A DUMMY VARIABLE TO READ IN THE LINE NUMBER (1-6)
ESITE1(J,L) - RANGE IN NMI OF LAUNCH BOUNDRY DATA PAIR
ESITE2(J,L) - ALTITUDE IN NMI OF LAUNCH BOUNDRY DATA PAIR

1, 5.00, 0.05
2, 5.10, 10.00
3, 15.00, 10.10
4, 20.00, 10.20
5, 20.10, 0.20
6, 5.00, 0.05è

THE SAM SYSTEM DATA BLOCKS WILL LOOK LIKE THESE BLOCKS BELOW. SINCE THERE ARE 9 SAM TYPES, THERE MUST BE A TOTAL OF 9 DATA BLOCKS.

6,20,10,30 SAM TYPE 2
1, 1.00, 0.10
2, 1.10, 10.00
3, 9.00, 10.10
4, 9.50, 10.20
5, 9.00, 0.20
6, 1.00, 0.10

5,30,2,5 SAM TYPE 3
1, 0.10, 0.10
2, 0.20, 9.00
3, 30.00, 9.10
4, 25.00, 0.20
5, 0.10, 0.10

6,100,5,10
1, 1.0, 1.0
2, 1.1, 10.0

```
3, 10.0, 10.1
4,  7.5,  7.5
5,  5.0,  5.0
6,  1.0,  1.0

6,100,5,10
1,  1.0,  1.0
2,  1.1, 10.0
3, 10.0, 10.1
4,  7.5,  7.5
5,  5.0,  5.0
6,  1.0,  1.0

6,100,5,10
1,  1.0,  1.0
2,  1.1, 10.0
3, 10.0, 10.1
4,  7.5,  7.5
5,  5.0,  5.0
6,  1.0,  1.0

6,100,5,10
1,  1.0,  1.0
2,  1.1, 10.0
3, 10.0, 10.1
4,  7.5,  7.5
5,  5.0,  5.0
6,  1.0,  1.0

6,100,5,10
1,  1.0,  1.0
2,  1.1, 10.0
3, 10.0, 10.1
4,  7.5,  7.5
5,  5.0,  5.0
6,  1.0,  1.0

6,100,5,10
1,  1.0,  1.0
2,  1.1, 10.0
3, 10.0, 10.1
4,  7.5,  7.5
5,  5.0,  5.0
6,  1.0,  1.0


FEND(L,1);L=1,7
```

START OF REACTIVE ARM FOOTPRINT DATA

FEND(L,1) - MIN RANGE BOUNDRY (IN NMI) FOR REACTIVE ARM FOR A SPECIFIC
- RANGE AND LAUNCH CONDITION. 7 VALUES REQUIRED ONE FOR EACH
- 15 DEGREES FROM 90 DEGREES TO THE LAUNCH AXIS TO ON AXIS.

4.00,4.50,5.00,7.00,7.50,8.00,8.50

SSEND(L,1);L=1,7

SSEND(L,1) - MAX RANGE BOUNDRY (IN NMI) FOR REACTIVE ARM FOR A SPECIFIC
- RANGE AND LAUNCH CONDITION. 7 VALUES REQUIRED ONE FOR EACH
- 15 DEGREES FROM 90 DEGREES TO THE LAUNCH AXIS TO ON AXIS.

19.60,19.70,20.90,22.20,24.80,28.60,33.00

KK - NUMBER OF LINES OF DATA TO READ THAT FOLLOW. DATA REPRESENTS EVERY
- TEN SECONDS OF TIME FOR REACTIVE ARM RANGE DATA SETS

5

FTPRTA(K,L);L=1,7

GROUND RANGE (IN FEET) OF REACTIVE ARM FOR EVERY TEN SECONDS OF FLIGHT

16000,17000,18000,19000,20000,22000,24000
26000,28000,30000,35000,40000,45000,50000
100000,105000,110000,115000,120000,125000,130000
135000,140000,145000,150000,155000,160000,165000
170000,175000,180000,185000,190000,195000,200000

ANGM,ALTMN

START OF ARM TRAJECTORY DATA

ANGM - HALF ANGLE (IN DEGREES) OF ARM SEEKER FOV
ALTMN - MIN. ALTITUDE (IN NMI) THAT ARM HAS TO BE BELOW BEFORE
- THE SEEKER WILL START SEARCHING FOR A SAM SITE

50,10.00

TARM(L)
ALTARM(L)

RNGARM(L)

THREE PARAMETERS (TIME, RANGE AND ALTITUDE) MATCH UP TO MAKE A DATA SET WHICH DEFINES THE ARM TRAJECTORY;10 DATA POINTS EXIST ON THREE LINES. EACH LINE CONTAINS ALL THE DATA FOR ONE PARAMETER.

TARM(L) - TIME (IN SECONDS) FROM LAUNCH OF ARM
ALTARM(L) - ALTITUDE (IN NMI) OF ARM AT GIVEN TIME FROM LAUNCH
RNGARM(L) - DOWNRANGE (IN NMI) OF ARM AT GIVEN TIME FROM LAUNCH

```
   0,   80,   90,  110,  130,  150,  170,  190,  210,  230
12.00, 11.00, 10.00, 9.00, 8.00, 7.00, 6.00, 5.00, 4.00,  0.0
   0,   5.0,  10.0, 15.0, 20.0, 25.0, 30.0, 32.0, 34.0, 36.0
```

ORDR
BM(L)

DATA FOR THE DIRECTIVE ARM TRAJECTORY; Nth ORDER AND COEFFICIENTS FOR THE CURVE INTERCEPT TIME = B0+B1*X+B2*X^2+...+BN*X^N

ORDR - NUMBER OF DATA VALUES TO READ IN MINUS ONE
BM(L) - ARRAY TO HOLD COEFFICIENTS

```
3
2.0,6.0,0.10,-0.01
```

HT(J),DIST(J)

SAM SITE TERRAIN MASKING DATA PAIRS; USED BY PROGRAM TO TERRAIN MASKING ONE LINE REQUIRED FOR EACH SAM SITE

HT(J) - HEIGHT OF OBSTRUCTING TERRAIN (IN NMI) AT DIST(J) FOR SITE J
DIST(J) - RANGE (IN NMI) FROM SITE LOCATION WHERE OBSTRUCTING TERRAIN
- IS LOCATED

```
0.03,4   SAM SITE TERRAIN MASKING DATA
0.03,4
0.03,4
```

CS(I),SIG(I)

PENETRATOR DATA RELATING TO RADAR CROSS-SECTION USED TO DETERMINE IF PENETRATOR IS DETECTABLE TO THE SAM SITES.
ONE LINE NEEDED FOR EACH PENETRATOR. FIRST LINE IS FOR A/C #1, ETC...

CS(I) - ?????
SIG(I) - RADAR CROSS-SECTION (IN SQUARE METERS) OF PENETRATOR

500,10 PENETRATOR CROSS-SECTION DATA
500,10
500,10
500,10

NEWSITES - # OF EW SITES THAT ARE PART OF THE SAM DATA BLOCK. MAX=9
EWID(9) - ID # OF SAM SITES THAT WILL BE EW SITES

IF NEWSITES = 0 AND IOP(3) < = 3 THEN NO MORE DATA WILL BE READ IN SO THE DATA FILE CAN END AFTER THE SINGLE ZERO. THIS IS THE CASE FOR NO IADS AND NO EW SITES.
IF NEWSITES > 0, MORE DATA IS NEEDED. IF A NON-IADS MODE IS SELECTED THEN THE VALUE OF NEWSITES MORE DATA IS NEEDED. IF AN IADS MODED IS SELECTED, MUCH MORE DATA IN NEEDED. WHEN NEWSITES > 0, THE FOLLOWING NUMBERS ARE READIN TO EWID(*) AND INDICATE THE ID NUMBER IDENTFYING THE SAM SITES AS EW SITES

3
1
2
19

EWPROCDLY(JJ),EWDAMDLY(JJ),EWDAMRAT(JJ)

DATA WHICH DEFINES THE DELAY THE EW SITES WILL HAVE BEFORE SENDING MESSAGES TO THE ADWOC CENTER AND EW REPORTING STATION ON DETECTIONS

EWPROCDLY(JJ) - NORMAL PROCESSING DELAY TIME (SECONDS) FOR THIS EW SITE
EWDAMDLY(JJ) - MAX. DELAY TIME (SECONDS) DUE TO MAX (PS=0.0) DAMAGE
EWDAMRAT(JJ) - DAMAGE DELAY SCALE FACTOR USED IN EQUATION COMPUTING
        - THE EXTRA DELAY DUE TO DAMAGE

10.0,100.0,1.0

EWPSMIN
EWPSMIN - MIN. PROBABILITY OF SURVIVAL FOR ALL EW SITE TO STAY IN SIM.

0.35

NUMRPT
RPTPROCDLY(JJ),RPTDAMDLY(JJ),RPTDAMRAT(JJ)
MINRPTPS

NEXT IS BLOCK OF DATA ASSOCIATED WITH THE EW REPORTING STATIONS.
THE FIRST IS THE TOTAL NUMBER OF STATIONS IN THIS SIMULATION, THE NEXT LINE DEFINES THE MESSAGE DELAY PARAMETERS, THE FINAL LINE IS THE MIN. PS

```
FOR THE REPORTING STATIONS
NUMRPT - MAX NUMBER OF EW REPORTING STATIONS IN THIS SIMULATION.
MINRPTPS - MIN. SURVIVAL PROB. FOR EW RPT STATION TO SAY IN SIM.

2               NUMRPT
10.0,100.0,1.0  RPTPROCDLY(JJ),RPTDAMDLY(JJ),RPTDAMRAT(JJ)
10.0,100.0,1.0
0.35            MINRPTPS

ADWPROCDLY,ADWDAMDLY,ADWDAMRAT
MINADWOCPS,PSADWOC

THE NEXT BLOCK RELATES TO THE ADWOC CENTER

ADWPROCDLY - NORMAL PROCESSING DELAY FROM DETECTION OF PENETRATOR
        - UNTIL ADWOC CENTER READY TO SEND MESSAGE (SECONDS)
ADWDAMDLY - MAX. EXTRA DELAY UNTIL ADWOC CENTER READY TO SEND
        - MESSAGE DUE TO SITE BEING DAMAGED
ADWDAMRAT - SCALE FACTOR USED WITH ADWDAMDLY(9) TO COMPUTE DELAY
        - DUE TO SITE DAMAGE
MINADWOCPS - MINIMUM SURVIVAL PROBABILITY FOR ADWOC TO SAY IN SIM.
PSADWOC - STARTING SURVIVAL PROBABILITY OF ADWOC

10.0,100.0,1.0  ADWPROCDLY,ADWDAMDLY,ADWDAMRAT
0.35,1.0        MINADWOCPS,PSADWOC

NUMBAT
BATPROCDLY(JJ),BATDAMDLY(JJ),BATDAMRAT(JJ)
MINBATPS

NEXT IS BLOCK OF DATA ASSOCIATED WITH THE BATTALION CENTERS.
THE FIRST IS THE TOTAL NUMBER OF CENTERS IN THIS SIMULATION, THE NEXT
LINE DEFINES THE MESSAGE DELAY PARAMETERS, THE FINAL LINE IS THE MIN. PS
FOR THE BATTALION CENTERS

2               NUMBAT
10.0,100.0,1.0  BATPROCDLY(JJ),BATDAMDLY(JJ),BATDAMRAT(JJ)
10.0,100.0,1.0
0.35            MINBATPS

THE FINAL BLOCK OF A SIMILAR NATURE IS FOR THE SAM SITES

SAMPROCDLY,SAMDAMDLY,SAMDAMRAT

10.0,100.0,1.0  SAMPROCDLY,SAMDAMDLY,SAMDAMRAT

THE NEXT BLOCK DEFINES THE IADS INTERCONNECTION. EACH IADS TYPE AND
UNIT NUMBER IS SPECIFIED AND THE CONNECTION TO ANOTHER IADS TYPE AND
UNIT NUMBER IS LISTED. THIS DEFINES THE POSSIBLE SENDING PATH FOR
MESSAGES.
```

NUMCON - # OF CONNECTIONS TO BE READIN IN IADS STRUCTURE DATA FILE
CONNECT(MAXCON,4) - HOLDS IADS CONNECTION STRUCTURE. INDICATES WHERE EACH IADS NODE CAN SEND A MESSAGE TO. FIRST PAIR OF #'S IDENTIFY SENDER AND SECOND PAIR IDENTIFY RECEIVER.
(*,1) - FROM TYPE # ;(*,2) - FROM UNIT #
(*,3) - TO TYPE # ;(*,4) - TO UNIT #
TYPE #1 - EW SITE; TYPE #2 - EW REPORTING STATION; TYPE #3 - ADWOC CENTER
TYPE #4 - BATTALION CENTER; TYPE #5 - SAM SITE, USE SAM ID # FOR UNIT #

38 NUMBER OF LINES TO FOLLOW DESCRIBING
1,1,2,1 CONNECTION DATA FOR IADS NETWORK
1,1,2,2
1,2,2,1
1,2,2,2
2,1,2,2
2,1,3,1
2,1,4,1
2,1,4,2
2,2,2,1
2,2,3,1
2,2,4,1
2,2,4,2
3,1,1,1
3,1,1,2
3,1,2,1
3,1,2,2
3,1,4,1
3,1,4,2
3,1,5,3
3,1,5,4
3,1,5,5
3,1,5,6
4,1,3,1
4,1,4,2
4,1,5,3
4,1,5,4
4,2,3,1
4,2,4,1
4,2,5,5
4,2,5,6
5,3,3,1
5,3,4,1
5,4,3,1
5,4,4,1
5,5,3,1
5,5,4,2
5,6,3,1
5,6,4,2

WHEN THE IADS ELEMENTS ARE TARGETTED IN THE SIMULATION, THEY ARE CONSIDERED NON-SAM TYPE TARGETS. THE TOTAL NUMBER OF NON-SAM TARGETS, NTARGETS, MUST REFLECT THIS CONDITION AND THE FOLLOWING LINES ARE ADDED TO GIVE THE SIMULATION DATA ON INDENTIFYING THE IADS TARGETS FROM OTHER NON-IADS TARGETS.

NUMIADSTGTS - TOTAL NUMBER OF IADS UNITS THAT ARE NON-SAM TARGETS
- IN THIS INPUT DATAFILE
IDIADSTGT(MAXIADSTGTS,2) - FOR EACH NON-SAM TARGET NUMBER, THIS
- HOLDS THE IADS TYPE AND UNIT NUMBER, USED TO IDENTIFY
- WHAT THE NON-SAM TARGET IS. (*,1) - IADS TYPE #;
- (*,2) - IADS UNIT #

5 NUMIADSTGTS
1,2,1 NON-SAM TGTNUM,IADS TYPE,IADS UNIT
2,2,2
3,3,1
4,4,1
5,4,2

APPENDIX C

The Input Data File for the

-25 Degree Appproach Angle

TEST85 FLIGHTPATH SET 8; -25 DEGREE APPROACH; IADS; FREE-FIRE

0,0,5,0,0,0,0,0,0,1 ** LABEL A **

12,6,0.4,1,5

0

** LABEL H **

-18.00,-10.00, 0.0, 0.3000, 1, 0.10, 8, 0.1, 1, 0.4000, 1.00, 6 EW # 1
-18.00, 10.00, 0.0, 0.3000, 1, 0.10, 8, 0.1, 1, 0.4000, 1.00, 6 EW # 2
-6.00, -5.00, 0.0, 0.3000, 2, 0.10, 1, 0.1, 1, 0.4000, 1.00, 16
0.00, 0.00, 0.0, 0.3000, 2, 0.10, 1, 0.1, 1, 0.4000, 1.00, 16
-11.00, 0.00, 0.0, 0.3000, 2, 0.10, 1, 0.1, 1, 0.4000, 1.00, 16
-6.00, 5.00, 0.0, 0.3000, 2, 0.10, 1, 0.1, 1, 0.4000, 1.00, 16

0 A/C #1 HITS EW SITE #1
0,1,0,0
2,7
4,0
-65.0,-10.2, 1.000, 0.125
-19.1,-10.2, 1.000, 0.125
-19.1, -8.2, 1.000, 0.125
-65.0, -8.2, 1.000, 0.125

0 A/C #2 HITS EW SITE #2
0,1,0,0
2,8
4,0
-65.0, 10.2, 1.000, 0.125
-19.1, 10.2, 1.000, 0.125
-19.1, 12.2, 1.000, 0.125
-65.0, 12.2, 1.000, 0.125

0 A/C #3 HITS SAM SITE #3
0,1,0,0
2,9
4,0
-58.9,-27.5, 1.000, 0.125
-7.1, -5.2, 1.000, 0.125
-7.1, -3.2, 1.000, 0.125
-65.0, -3.2, 1.000, 0.125

```
0    A/C #4 HITS SAM SITE #4
0,1,0,0
2,10
4,0
-58.9,-27.5,  1.000, 0.125
 -1.1,  0.2,  1.000, 0.125
 -1.1,  2.2,  1.000, 0.125
-65.0,  2.2,  1.000, 0.125

0    A/C #5 HITS SAM SITE #5
0,1,0,0
2,11
4,0
-58.9,-27.5,  1.000, 0.125
-12.1,  0.2,  1.000, 0.125
-12.1,  2.2,  1.000, 0.125
-65.0,  2.2,  1.000, 0.125

0    A/C #6 HITS SAM SITE #6
0,1,0,0
2,12
4,0
-58.9,-27.5,  1.000, 0.125
 -7.1,  5.2,  1.000, 0.125
 -7.1,  7.2,  1.000, 0.125
-65.0,  7.2,  1.000, 0.125

1    WEAPON #1 FROM A/C #1 TO HIT EW SITE #1
1,1
2
-19.10,-10.20,  1.000, 0.30
-18.00,-10.00,  0.000, 0.30

1    WEAPON #2 FROM A/C #2 TO HIT EW SITE #2
1,2
2
-19.10, 10.20,  1.000, 0.30
-18.00, 10.00,  0.000, 0.30

1    WEAPON #3 FROM A/C #3 TO HIT SAM SITE #3
1,3
2
 -7.10, -5.20,  1.000, 0.30
```

```
 -6.00, -5.00,  0.000, 0.30

1    WEAPON #4 FROM A/C #4 TO HIT SAM SITE #4
1,4
2
  1.10,  0.20,  1.000, 0.30
  0.00,  0.00,  0.000, 0.30

1    WEAPON #5 FROM A/C #5 TO HIT SAM SITE #5
1,5
2
-12.10,  0.20,  1.000, 0.30
-11.00,  0.00,  0.000, 0.30

1    WEAPON #6 FROM A/C #6 TO HIT SAM SITE #6
1,6
2
 -7.10,  5.20,  1.000, 0.30
 -6.00,  5.00,  0.000, 0.30

0.5,0.2,0.3,0.4,0.5,0.6,0.7,0.1,0.9 STANDOFF WPNS PK FOR SAM TYPES
0.1,0.2,0.3,0.4,0.5 STANDOFF WPNS PK FOR 5 NON-SAM TYPES

1,1,1,1,1,1
1,1,1,1,1,1
1,1,1,1,1,1
1,1,1,1,1,1
1,1,1,1,1,1
1,1,1,1,1,1
1,1,1,1,1,1
1,1,1,1,1,1
1,1,1,1,1,1
1,1,1,1,1,1
1,1,1,1,1,1
1,1,1,1,1,1

0,30


0.35

5,10,5,10 SAM TYPE 1 - ALL SAMS ARE THIS TYPE
1, 0.10,  0.10
```

2, 0.20, 5.00
3, 5.00, 5.10
4, 5.10, 0.50
5, 0.10, 0.10

6,20,5,10 type 2
1,1.00,0.100
2,1.01,10.00
3,9.00,10.10
4,9.50,10.20
5,9.00,1.00
6,1.00,0.100

5,30,5,10 SAM TYPE 3
1, 0.100, 0.100
2, 0.200, 9.00
3, 30.00, 9.1
4, 25.00, 0.15
5, 0.100, 0.10

6,100,5,10 SAM TYPE 4
1,1.0, 1.0
2, 1.1, 10.0
3,10.0, 10.1
4, 7.5, 7.5
5, 5.0, 5.0
6, 1.0, 1.0

6,100,5,10 SAM TYPE 5
1, 1.0, 1.0
2, 1.1, 10.0
3,10.0, 10.1
4, 7.5, 7.5
5, 5.0, 5.0
6, 1.0, 1.0

6,100,5 10 SAM TYPE 6
1, 1.0, 1.0
2, 1.1, 10.0
3,10.0, 10.1
4, 7.5, 7.5
5, 5.0, 5.0
6, 1.0, 1.0

```
6,100,5,10  SAM TYPE 7
1, 1.0,  1.0
2, 1.1, 10.0
3,10.0, 10.1
4, 7.5,  7.5
5, 5.0,  5.0
6, 1.0,  1.0

5,100,5,10  SAM TYPE 8 - ALL EW SITES ARE THIS TYPE
1, 0.01,  0.01
2, 0.11, 10.0
3,30.1,  10.1
4,30.0,   0.1
5, 0.01,  0.01

6,100,5,10  SAM TYPE 9
1, 1.0,  1.0
2, 1.1, 10.0
3,10.0, 10.1
4, 7.5,  7.5
5, 5.0,  5.0
6, 1.0,  1.0

4.00,4.50,5.00,7.00,7.50,8.00,8.50
19.60,19.70,20.90,22.20,24.80,28.60,33.00

5
16000,17000,18000,19000,20000,22000,24000
26000,28000,30000,35000,40000,45000,50000
100000,105000,110000,115000,120000,125000,130000
135000,140000,145000,150000,155000,160000,165000
170000,175000,180000,185000,190000,195000,200000

50,10.00

   0,   80,   90,  110,  130,  150,  170,  190,  210,  230
12.00, 11.00, 10.00, 9.00, 8.00, 7.00, 6.00, 5.00, 4.00,  0.0
   0,   5.0,  10.0, 15.0, 20.0, 25.0, 30.0, 32.0, 34.0, 36.0


3
2.0,6.0,0.10,-0.01
```

```
0.03,4
0.03,4
0.03,4
0.03,4
0.03,4
0.03,4

500,10
500,10
500,10
500,10
500,10
500,10
500,10
500,10
500,10
500,10
500,10
500,10

2  TOTAL # OF EW SITES                          ** LABEL B **
1  SAM SITE #1 IS AN EW SITE #1
2  SAM SITE #2 IS AN EW SITE #2
                                          ** LABEL C **
355.0,100.0,1.0 EWPROCDLY(JJ),EWDAMDLY(JJ),EWDAMRAT(JJ)
355.0,100.0,1.0

0.35  EWPSMIN

2              NUMRPT
10.0,100.0,1.0   RPTPROCDLY(JJ),RPTDAMDLY(JJ),RPTDAMRAT(JJ)
10.0,100.0,1.0
0.35             MINRPTPS

10.0,100.0,1.0   ADWPROCDLY,ADWDAMDLY,ADWDAMRAT
0.35,1.0         MINADWOCPS,PSADWOC              ** LABEL D **

2              NUMBAT
10.0,100.0,1.0   BATPROCDLY(JJ),BATDAMDLY(JJ),BATDAMRAT(JJ)
10.0,100.0,1.0
0.35             MINBATPS
```

```
10.0,100.0,1.0  SAMPROCDLY,SAMDAMDLY,SAMDAMRAT   ** LABEL E **

46      NUMBER OF LINES TO FOLLOW DESCRIBING      ** LABEL F **
1,1,2,1 CONNECTION DATA FOR IADS NETWORK
1,1,2,2
1,1,3,1
1,2,2,1
1,2,2,2
1,2,3,1
2,1,1,1
2,1,1,2
2,1,2,2
2,1,3,1
2,1,4,1
2,1,5,3
2,1,5,4
2,2,1,1
2,2,1,2
2,2,2,1
2,2,3,1
2,2,4,2
2,2,5,5
2,2,5,6
3,1,1,1
3,1,1,2
3,1,2,1
3,1,2,2
3,1,4,1
3,1,4,2
3,1,5,3
3,1,5,4
3,1,5,5
3,1,5,6
4,1,3,1
4,1,4,2
4,1,5,3
4,1,5,4
4,2,3,1
4,2,4,1
4,2,5,5
4,2,5,6
5,3,3,1
5,3,4,1
```

5,4,3,1
5,4,4,1
5,5,3,1
5,5,4,2
5,6,3,1
5,6,4,2

5 NUMIADSTGTS ** LABEL G **
1,2,1 NON-SAM TGTNUM,IADS TYPE,IADS UNIT
2,2,2
3,3,1
4,4,1
5,4,2

APPENDIX D

A Sample TACOPS

Output Data File

```
STANDARD TACOPS OUTPUT WITH DEBUG SET = 1;
TEST85 FLIGHTPATH SET 8; -25 DEGREE APPROACH; IADS; FREE-FIRE
 STARTING ISEED=      1000000.00 RANDOM NO.=*********
   1. SEC PEN.  1 CHANGES COURSE 270.0=AZ    0.0=EL   -64.87  -10.20    1.00
   1. SEC PEN.  2 CHANGES COURSE 270.0=AZ    0.0=EL   -64.87   10.20    1.00
   1. SEC PEN.  3 CHANGES COURSE 246.7=AZ    0.0=EL   -58.79  -27.45    1.00
   1. SEC PEN.  4 CHANGES COURSE 244.4=AZ    0.0=EL   -58.79  -27.45    1.00
   1. SEC PEN.  5 CHANGES COURSE 239.4=AZ    0.0=EL   -58.79  -27.44    1.00
   1. SEC PEN.  6 CHANGES COURSE 237.7=AZ    0.0=EL   -58.79  -27.43    1.00
   1. SEC ADWOC WILL SEND START UP MESSAGE AT TIME  10. TO EW SITE #  1
        MSGID=          1
   1. SEC ADWOC WILL SEND START UP MESSAGE AT TIME  10. TO EW SITE #  1
         VIA RPT STA #         1  MSGID=          1
   1. SEC ADWOC WILL SEND START UP MESSAGE AT TIME  10. TO EW SITE #  2
        MSGID=          2
   1. SEC ADWOC WILL SEND START UP MESSAGE AT TIME  10. TO EW SITE #  2
         VIA RPT STA #         1  MSGID=          2
  10. SEC EW SITE  1 ORDERED TO TURN ON; DOES SO.
  10. SEC RPT STA  1 WILL SEND MESSAGE AT TIME   20. MSGID #  1
  10. SEC RPT STA  2 WILL SEND MESSAGE AT TIME   20. MSGID #  1
  10. SEC EW SITE  2 ORDERED TO TURN ON; DOES SO.
  10. SEC RPT STA  1 WILL SEND MESSAGE AT TIME   20. MSGID #  2
  10. SEC RPT STA  2 WILL SEND MESSAGE AT TIME   20. MSGID #  2
 116. SEC EW SITE  1 DETECTS PEN.  3    -45.58   -21.77     1.00
 116. SEC EW SITE  1 DETECTS PEN.  4    -45.82   -21.23     1.00
 116. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  471.
 118. SEC EW SITE  1 DETECTS PEN.  5    -46.21   -19.99     1.00
 118. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  473.
 119. SEC EW SITE  1 DETECTS PEN.  6    -46.32   -19.56     1.00
 119. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  474.
 136. SEC EW SITE  1 DETECTS PEN.  1    -48.00   -10.20     1.00
 136. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  491.
 136. SEC EW SITE  2 DETECTS PEN.  2    -48.00    10.20     1.00
 136. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  491.
 199. SEC EW SITE  1 DETECTS PEN.  2    -40.12    10.20     1.00
 199. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  554.
 199. SEC EW SITE  2 DETECTS PEN.  1    -40.12   -10.20     1.00
 199. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  554.
 211. SEC EW SITE  2 DETECTS PEN.  6    -36.60   -13.42     1.00
 211. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  566.
 213. SEC EW SITE  2 DETECTS PEN.  5    -35.99   -13.94     1.00
 213. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  568.
 223. SEC EW SITE  2 DETECTS PEN.  4    -33.76   -15.45     1.00
 223. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  578.
 229. SEC EW SITE  2 DETECTS PEN.  3    -32.61   -16.18     1.00
 229. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  584.
 367. SEC A/C   1 LAUNCHES PEN. #  7 (TYPE 1 WEAPON) ;A/C PS =1.00
 367. SEC A/C   2 LAUNCHES PEN. #  8 (TYPE 1 WEAPON) ;A/C PS =1.00
 368. SEC PEN.  1 CHANGES COURSE 180.0=AZ    0.0=EL   -19.10  -10.10    1.00
 368. SEC PEN.  2 CHANGES COURSE 180.0=AZ    0.0=EL   -19.10   10.30    1.00
```

```
368. SEC PEN.  7 CHANGES COURSE 259.7=AZ  -41.8=EL   -18.92  -10.17    0.84
368. SEC PEN.  8 CHANGES COURSE 280.3=AZ  -41.8=EL   -18.92   10.17    0.84
368. SEC EW SITE  1 DETECTS PEN.  7    -18.92   -10.17     0.84
368. SEC EW SITE  1 DETECTS PEN.  8    -18.92    10.17     0.84
368. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  723.
368. SEC EW SITE  2 DETECTS PEN.  7    -18.92   -10.17     0.84
368. SEC EW SITE  2 DETECTS PEN.  8    -18.92    10.17     0.84
368. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  723.
372. SEC EW SITE  1 LOST DETECTION PEN.  8    -18.04    10.01     0.04
372. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  727.
372. SEC EW SITE  2 LOST DETECTION PEN.  7    -18.04   -10.01     0.04
372. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  727.
373. SEC STANDOFF WEAPON  7 IMPACTS SITE  1 PS=0.90 RND#**** -18.0 -10.0
  IMPACTED SITE IS EW SITE #  1
373. SEC STANDOFF WEAPON  8 IMPACTS SITE  2 PS=0.90 RND#**** -18.0  10.0
  IMPACTED SITE IS EW SITE #  2
373. SEC EW SITE  1 LOST DETECTION PEN.  7    -18.04   -10.01     0.04
373. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  738.
373. SEC EW SITE  2 LOST DETECTION PEN.  8    -18.04    10.01     0.04
373. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  738.
384. SEC PEN.  1 CHANGES COURSE  90.0=AZ    0.0=EL   -19.20   -8.20    1.00
384. SEC PEN.  2 CHANGES COURSE  90.0=AZ    0.0=EL   -19.20   12.20    1.00
435. SEC A/C   5 LAUNCHES PEN. # 11 (TYPE 1 WEAPON) ;A/C PS =1.00
436. SEC PEN.  5 CHANGES COURSE 180.0=AZ    0.0=EL   -12.10    0.32    1.00
436. SEC PEN. 11 CHANGES COURSE 280.3=AZ  -41.8=EL   -11.89    0.16    0.81
436. SEC EW SITE  1 DETECTS PEN. 11    -11.89     0.16     0.81
436. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  801.
436. SEC EW SITE  2 DETECTS PEN. 11    -11.89     0.16     0.81
436. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  801.
440. SEC EW SITE  1 LOST DETECTION PEN. 11    -11.01     0.00     0.01
440. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  805.
440. SEC EW SITE  2 LOST DETECTION PEN. 11    -11.01     0.00     0.01
440. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  805.
441. SEC STANDOFF WEAPON 11 IMPACTS SITE  5 PS=0.50 RND#**** -11.0   0.0
441. SEC STANDOFF WEAPON 11 IMPACTS SAM SITE # 5 PS=0.50 RND#0.00
451. SEC A/C   3 LAUNCHES PEN. #  9 (TYPE 1 WEAPON) ;A/C PS =1.00
452. SEC PEN.  3 CHANGES COURSE 180.0=AZ    0.0=EL    -7.10   -5.10    1.00
452. SEC PEN.  5 CHANGES COURSE  90.0=AZ    0.0=EL   -12.22    2.20    1.00
452. SEC PEN.  9 CHANGES COURSE 259.7=AZ  -41.8=EL    -6.92   -5.17    0.83
452. SEC EW SITE  1 DETECTS PEN.  9     -6.92    -5.17     0.83
452. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  817.
452. SEC EW SITE  2 DETECTS PEN.  9     -6.92    -5.17     0.83
452. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  817.
456. SEC EW SITE  1 LOST DETECTION PEN.  9     -6.04    -5.01     0.03
456. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  821.
456. SEC EW SITE  2 LOST DETECTION PEN.  9     -6.04    -5.01     0.03
456. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  821.
457. SEC STANDOFF WEAPON  9 IMPACTS SITE  3 PS=0.50 RND#****  -6.0  -5.0
457. SEC STANDOFF WEAPON  9 IMPACTS SAM SITE # 3 PS=0.50 RND#0.00
468. SEC PEN.  3 CHANGES COURSE  90.0=AZ    0.0=EL    -7.20   -3.20    1.00
```

```
481. SEC RPT STA  1 WILL SEND MESSAGE AT TIME  491. MSGID #  3
481. SEC RPT STA  2 WILL SEND MESSAGE AT TIME  491. MSGID #  3
481. SEC SITE  3 NO LOCK-ON  PEN.  3 ;BAD FUZE ANGLE AT INTERCEPT
                                  -8.83    -3.20      1.00
481. SEC SITE  3 STARTS RADIATING
481. SEC SITE  3 LOCKS ON PEN.  4       -4.68      -1.52       1.00
481. SEC SITE  4 STARTS RADIATING
481. SEC SITE  4 LOCKS ON PEN.  4       -4.68      -1.52       1.00
481. SEC SITE  5 STARTS RADIATING
481. SEC SITE  5 LOCKS ON PEN.  3       -8.83      -3.20       1.00
483. SEC RPT STA  1 WILL SEND MESSAGE AT TIME  493. MSGID #  4
483. SEC RPT STA  2 WILL SEND MESSAGE AT TIME  493. MSGID #  4
484. SEC RPT STA  1 WILL SEND MESSAGE AT TIME  494. MSGID #  5
484. SEC RPT STA  2 WILL SEND MESSAGE AT TIME  494. MSGID #  5
484. SEC SITE  6 STARTS RADIATING
484. SEC SITE  6 LOCKS ON PEN.  6       -7.74       4.80       1.00
490. SEC A/C   6 LAUNCHES PEN. # 12 (TYPE 1 WEAPON) ;A/C PS =1.00
491. SEC PEN.  6 CHANGES COURSE 180.0=AZ    0.0=EL    -7.10     5.32     1.00
491. SEC PEN. 12 CHANGES COURSE 280.3=AZ  -41.8=EL    -6.89     5.16     0.81
491. SEC EW SITE  1 DETECTS PEN. 12     -6.89      5.16      0.81
491. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  856.
491. SEC EW SITE  2 DETECTS PEN. 12     -6.89      5.16      0.81
491. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  856.
491. SEC SITE  3 FIRES ON PEN.  4 SALVO   1       -3.55     -0.98      1.00
491. SEC SITE  4 FIRES ON PEN.  4 SALVO   1       -3.55     -0.98      1.00
491. SEC SITE  5 FIRES ON PEN.  3 SALVO   1      -10.08     -3.20      1.00
494. SEC SITE  6 FIRES ON PEN.  6 SALVO   1       -7.10      5.69      1.00
495. SEC EW SITE  1 LOST DETECTION PEN. 12      -6.01      5.00      0.01
495. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  860.
495. SEC EW SITE  2 LOST DETECTION PEN. 12      -6.01      5.00      0.01
495. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  860.
496. SEC STANDOFF WEAPON 12 IMPACTS SITE  6 PS=0.50 RND#****  -6.0    5.0
496. SEC STANDOFF WEAPON 12 IMPACTS SAM SITE # 6 PS=0.50 RND#0.00
501. SEC RPT STA  1 WILL SEND MESSAGE AT TIME  511. MSGID #  6
501. SEC RPT STA  2 WILL SEND MESSAGE AT TIME  511. MSGID #  6
501. SEC RPT STA  1 WILL SEND MESSAGE AT TIME  511. MSGID #  7
501. SEC RPT STA  2 WILL SEND MESSAGE AT TIME  511. MSGID #  7
501. SEC SITE  4 INTERCEPTS PEN.  4    -2.4    -0.4 NEW PS  =  0.90
502. SEC SITE  6 INTERCEPTS PEN.  6    -7.1     6.7 NEW PS  =  0.95
503. SEC SITE  4 FIRES ON PEN.  4 SALVO   2       -2.20     -0.33      1.00
503. SEC SITE  5 INTERCEPTS PEN.  3   -11.6    -3.2 NEW PS  =  0.95
504. SEC SITE  6 FIRES ON PEN.  6 SALVO   2       -7.10      6.94      1.00
505. SEC SITE  5 FIRES ON PEN.  3 SALVO   2      -11.83     -3.20      1.00
507. SEC PEN.  6 CHANGES COURSE  90.0=AZ    0.0=EL    -7.22     7.20     1.00
510. SEC SITE  4 INTERCEPTS PEN.  4    -1.4     0.1 NEW PS  =  0.81
512. SEC A/C   4 LAUNCHES PEN. # 10 (TYPE 1 WEAPON) ;A/C PS =0.81
512. SEC SITE  4 FIRES ON PEN.  4 SALVO   3       -1.19      0.16      1.00
513. SEC PEN.  4 CHANGES COURSE 180.0=AZ    0.0=EL    -1.10     0.23     1.00
513. SEC PEN. 10 CHANGES COURSE  79.7=AZ  -41.8=EL     1.05     0.19     0.95
513. SEC EW SITE  1 DETECTS PEN. 10      1.05      0.19      0.95
```

```
513. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  878.
513. SEC EW SITE  2 DETECTS PEN. 10      1.05     0.19     0.95
513. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  878.
517. SEC EW SITE  1 LOST DETECTION PEN. 10      0.17     0.03     0.15
517. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  882.
517. SEC EW SITE  2 LOST DETECTION PEN. 10      0.17     0.03     0.15
517. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  882.
517. SEC SITE  3 MISSES PEN.  4 > MAX RANGE    -1.10    0.73    1.00
CD1-CD4:    460.16   482.15 24513.00 24513.00
517. SEC SITE  4 INTERCEPTS PEN.  4    -1.1     0.7 NEW PS = 0.73
517. SEC SITE  6 INTERCEPTS PEN.  6    -8.5     7.2 NEW PS = 0.90
518. SEC STANDOFF WEAPON 10 IMPACTS SITE  4 PS=0.60 RND#****   0.0   0.0
518. SEC STANDOFF WEAPON 10 IMPACTS SAM SITE # 4 PS=0.60 RND#0.00
518. SEC SITE  3 STOPS RADIATING
519. SEC SITE  4 FIRES ON PEN.  4 SALVO   4       -1.10     0.98     1.00
519. SEC SITE  6 STARTS RADIATING
519. SEC SITE  6 LOCKS ON PEN.  6       -8.72       7.20       1.00
520. SEC SITE  5 INTERCEPTS PEN.  3   -13.7    -3.2 NEW PS = 0.90
521. SEC SITE  5 STOPS RADIATING
528. SEC SITE  4 INTERCEPTS PEN.  4    -1.1     2.1 NEW PS = 0.69
529. SEC PEN.  4 CHANGES COURSE  90.0=AZ    0.0=EL    -1.13    2.20    1.00
529. SEC SITE  4 NO LOCK-ON  PEN.  4 ;BAD FUZE ANGLE AT INTERCEPT
                                -1.13     2.20     1.00
529. SEC SITE  4 STOPS RADIATING
529. SEC SITE  6 FIRES ON PEN.  6 SALVO   3       -9.97     7.20     1.00
530. SEC SITE  4 STARTS RADIATING
530. SEC SITE  4 LOCKS ON PEN.  4       -1.26       2.20       1.00
536. SEC EW SITE  1 LOST DETECTION PEN.  2    -38.20    12.20     1.00
536. SEC EW SITE  1 WILL SEND MESSAGE AT TIME  901.
540. SEC SITE  4 FIRES ON PEN.  4 SALVO   5       -2.51     2.20     1.00
554. SEC SITE  6 MISSES PEN.  6 > MAX RANGE   -13.09    7.20    1.00
CD1-CD4:    460.55   533.98 24507.00 24507.00
555. SEC SITE  6 NO LOCK-ON  PEN.  6 ;BAD FUZE ANGLE AT INTERCEPT
                               -13.22     7.20     1.00
556. SEC SITE  6 STARTS RADIATING
556. SEC SITE  6 LOCKS ON PEN.  4       -4.51       2.20       1.00
558. SEC SITE  4 MISSES PEN.  4 > MAX RANGE    -4.76    2.20    1.00
CD1-CD4:    483.24   556.67 24529.00 24529.00
559. SEC SITE  4 NO LOCK-ON  PEN.  4 ;BAD FUZE ANGLE AT INTERCEPT
                               -4.88     2.20     1.00
559. SEC SITE  4 STOPS RADIATING
562. SEC SITE  5 STARTS RADIATING
562. SEC SITE  5 LOCKS ON PEN.  4       -5.26       2.20       1.00
564. SEC RPT STA  1 WILL SEND MESSAGE AT TIME  574. MSGID #  8
564. SEC RPT STA  2 WILL SEND MESSAGE AT TIME  574. MSGID #  8
564. SEC RPT STA  1 WILL SEND MESSAGE AT TIME  574. MSGID #  9
564. SEC RPT STA  2 WILL SEND MESSAGE AT TIME  574. MSGID #  9
566. SEC EW SITE  2 LOST DETECTION PEN.  1    -41.95    -8.20     1.00
566. SEC EW SITE  2 WILL SEND MESSAGE AT TIME  931.
566. SEC SITE  6 FIRES ON PEN.  4 SALVO   4       -5.76     2.20     1.00
```

572. SEC SITE 5 FIRES ON PEN. 4 SALVO 3 -6.51 2.20 1.00
576. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 586. MSGID # 10
576. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 586. MSGID # 10
577. SEC SITE 6 INTERCEPTS PEN. 4 -7.1 2.2 NEW PS = 0.65
578. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 588. MSGID # 11
578. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 588. MSGID # 11
579. SEC SITE 6 FIRES ON PEN. 4 SALVO 5 -7.38 2.20 1.00
585. SEC SITE 5 INTERCEPTS PEN. 4 -8.1 2.2 NEW PS = 0.62
587. SEC SITE 5 FIRES ON PEN. 4 SALVO 4 -8.38 2.20 1.00
588. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 598. MSGID # 12
588. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 598. MSGID # 12
594. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 604. MSGID # 13
594. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 604. MSGID # 13
594. SEC SITE 6 INTERCEPTS PEN. 4 -9.3 2.2 NEW PS = 0.59
595. SEC SITE 6 STOPS RADIATING
597. SEC SITE 5 INTERCEPTS PEN. 4 -9.6 2.2 NEW PS = 0.56
599. SEC SITE 5 FIRES ON PEN. 4 SALVO 5 -9.88 2.20 1.00
608. SEC SITE 5 INTERCEPTS PEN. 4 -11.0 2.2 NEW PS = 0.53
610. SEC SITE 5 FIRES ON PEN. 4 SALVO 6 -11.26 2.20 1.00
614. SEC EW SITE 2 LOST DETECTION PEN. 2 -47.95 12.20 1.00
614. SEC EW SITE 2 WILL SEND MESSAGE AT TIME 979.
615. SEC EW SITE 1 LOST DETECTION PEN. 1 -48.08 -8.20 1.00
615. SEC EW SITE 1 WILL SEND MESSAGE AT TIME 980.
620. SEC SITE 5 INTERCEPTS PEN. 4 -12.5 2.2 NEW PS = 0.50
622. SEC SITE 5 FIRES ON PEN. 4 SALVO 7 -12.76 2.20 1.00
637. SEC SITE 5 INTERCEPTS PEN. 4 -14.6 2.2 NEW PS = 0.48
638. SEC SITE 5 NO LOCK-ON PEN. 4 ;BAD FUZE ANGLE AT INTERCEPT
-14.76 2.20 1.00
638. SEC SITE 5 STOPS RADIATING
718. SEC EW SITE 1 LOST DETECTION PEN. 5 -45.47 2.20 1.00
718. SEC EW SITE 1 WILL SEND MESSAGE AT TIME 1083.
731. SEC EW SITE 2 LOST DETECTION PEN. 5 -47.09 2.20 1.00
731. SEC EW SITE 2 WILL SEND MESSAGE AT TIME 1096.
733. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 743. MSGID # 14
733. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 743. MSGID # 14
733. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 743. MSGID # 15
733. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 743. MSGID # 15
737. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 747. MSGID # 16
737. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 747. MSGID # 16
737. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 747. MSGID # 17
737. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 747. MSGID # 17
738. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 748. MSGID # 18
738. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 748. MSGID # 18
738. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 748. MSGID # 19
738. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 748. MSGID # 19
770. SEC EW SITE 2 LOST DETECTION PEN. 3 -44.95 -3.20 1.00
770. SEC EW SITE 2 WILL SEND MESSAGE AT TIME 1135.
789. SEC EW SITE 1 LOST DETECTION PEN. 3 -47.33 -3.20 1.00
789. SEC EW SITE 1 WILL SEND MESSAGE AT TIME 1154.
790. SEC EW SITE 1 LOST DETECTION PEN. 6 -42.59 7.20 1.00

790. SEC EW SITE 1 WILL SEND MESSAGE AT TIME 1155.
801. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 811. MSGID # 20
801. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 811. MSGID # 20
801. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 811. MSGID # 21
801. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 811. MSGID # 21
805. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 815. MSGID # 22
805. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 815. MSGID # 22
805. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 815. MSGID # 23
805. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 815. MSGID # 23
817. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 827. MSGID # 24
817. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 827. MSGID # 24
817. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 827. MSGID # 25
817. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 827. MSGID # 25
821. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 831. MSGID # 26
821. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 831. MSGID # 26
821. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 831. MSGID # 27
821. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 831. MSGID # 27
833. SEC EW SITE 2 LOST DETECTION PEN. 6 -47.97 7.20 1.00
833. SEC EW SITE 2 WILL SEND MESSAGE AT TIME 1198.
856. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 866. MSGID # 28
856. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 866. MSGID # 28
856. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 866. MSGID # 29
856. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 866. MSGID # 29
860. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 870. MSGID # 30
860. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 870. MSGID # 30
860. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 870. MSGID # 31
860. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 870. MSGID # 31
878. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 888. MSGID # 32
878. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 888. MSGID # 32
878. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 888. MSGID # 33
878. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 888. MSGID # 33
882. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 892. MSGID # 34
882. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 892. MSGID # 34
882. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 892. MSGID # 35
882. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 892. MSGID # 35
884. SEC EW SITE 1 LOST DETECTION PEN. 4 -45.51 2.20 1.00
884. SEC EW SITE 1 WILL SEND MESSAGE AT TIME 1249.
896. SEC EW SITE 2 LOST DETECTION PEN. 4 -47.01 2.20 1.00
896. SEC EW SITE 2 WILL SEND MESSAGE AT TIME 1261.
901. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 911. MSGID # 36
901. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 911. MSGID # 36
931. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 941. MSGID # 37
931. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 941. MSGID # 37
979. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 989. MSGID # 38
979. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 989. MSGID # 38
980. SEC RPT STA 1 WILL SEND MESSAGE AT TIME 990. MSGID # 39
980. SEC RPT STA 2 WILL SEND MESSAGE AT TIME 990. MSGID # 39
--
SITE 1 HAS 6 SALVOS LEFT OF 6 PS 0.9000 -18.0 -10.0
SITE 2 HAS 6 SALVOS LEFT OF 6 PS 0.9000 -18.0 10.0

```
    SITE  3 HAS  15 SALVOS LEFT OF  16 PS 0.5000  -6.0  -5.0
    SITE  4 HAS  11 SALVOS LEFT OF  16 PS 0.5950   0.0   0.0
    SITE  5 HAS   9 SALVOS LEFT OF  16 PS 0.5000 -11.0   0.0
    SITE  6 HAS  11 SALVOS LEFT OF  16 PS 0.5000  -6.0   5.0

TOTAL SITE DAMAGE BEFORE THIS RUN =    0.000
TOTAL SITE DAMAGE TO THE PRESENT  =    2.105
TOTAL SITE DAMAGE DURING THIS RUN =    2.105

      AIRCRAFT DAMAGE BY SAMS =    0.716      ** LABEL A **
 SITE DAMAGE BY NON-ARM WEAPONS =    2.105
SITE DAMAGE BY PRE-EMPTIVE ARMS =    0.000
  SITE DAMAGE BY REACTIVE ARMS =    0.000
NON-SAM GROUND TARGET  1 HAS FINAL PS = 1.0000
NON-SAM GROUND TARGET  2 HAS FINAL PS = 1.0000
NON-SAM GROUND TARGET  3 HAS FINAL PS = 1.0000
NON-SAM GROUND TARGET  4 HAS FINAL PS = 1.0000
NON-SAM GROUND TARGET  5 HAS FINAL PS = 1.0000
```

APPENDIX E

LISTING OF FORTRAN CODE

AND OTHER FILES

The programs are separated from each other by a long line of dashes;
for example: "----------"

FIRST THE NEW IADS MODEL FORTRAN PROGRAMS ARE LISTED IN THE FOLLOWING ORDER:

CDELAY
DELMES
DLYMES
EWDET
EWDET2
EWMES
FIRE
FIRE0
FIRE1
FIRE2
FIRE3
FIRE4
GETDET
HOLDMES
IADS
IADSPS
INITEW
MESRDY
NSPS
PSEW0
RADWOC
RBAT
REDIAD
REW
REWSTA
RSAM
SETID
SHIFTU

THEN THE IADS MODEL FORTRAN SUPPORT PROGRAMS ARE LISTED IN THIS ORDER:

EWOFF
MESSUM
SAMOFF
SAMON

THEN THE ORIGINAL TACOPS PROGRAMS MODIFED FOR THE IADS MODEL ARE LISTED IN THIS ORDER:

REDIN
TACOPS
TICK2
VARIBL

FINALLY THE CONTENTS OF THE 'DEF' FILES USED IN THE IADS PROGRAMS ARE LISTED IN THIS ORDER:

ADWDATA DEF
ADWMES1 DEF
BATDATA DEF
BATMES1 DEF
CON1 DEF
EWDATA DEF
EWMES2 DEF
EW1 DEF
HOLDM1 DEF
KNOWN1 DEF
MESID DEF
QIADS DEF
RPTDATA DEF
RPTMES1 DEF
SAMDATA DEF
SAMMES1 DEF

The listing of the program starts now.

```
      SUBROUTINE CDELAY(PROCDLY,DAMGDLY,DAMGRAT,PS,DELAY)

C.....LATEST CHANGE MADE  7/01/91 BFS.....
C     COMPUTES THE DELAY FOR PROCESSING DATA BASED UPON NORMAL TIMES
C     AND ADDITIONAL DELAY DUE TO DAMAGE OF THE SITE
```

```
C
C     CALLED BY - EWDET
C     CALLS     - NONE
C.....................................................................
C
C      PASSED PARAMETERS
C
C      PROCDLY - NORMAL PROCESSING DELAY TIME FOR THIS UNIT
C      DAMGDLY - MAX. DELAY TIME (SECONDS) FOR MAX. DAMAGE
C      DAMGRAT - DAMAGE RATIO; SCALE FACTOR USED IN DELAY COMP.
C      PS - PROB. OF SURVIVAL OF UNIT
C      DELAY - COMPUTED DELAY OF MESSAGE PROCESSING TIME BASED
C            - UPON PASSED INPUTS

      REAL DAMGRAT,PS,DELAY,DAMGDLY
C.....................................................................

C ... LINEAR DAMAGE DELAY FUNCTION ...

      DELAY=DAMGDLY*((1.0-PS)/DAMGRAT)

C ... EXPONENTIAL DAMAGE DELAY FUNCTION ...

C     DELAY=DAMGDLY*(1-EXP(-(1.0-PS)/DAMGRAT))

      DELAY=PROCDLY+DELAY

      END
----------------------------------------------------------------------
      SUBROUTINE DELMES(TYPE,UNIT)

C.....LATEST CHANGE MADE 07/27/91 BFS.....
C     LOCATES ALL MESSAGES WAITING TO BE SENT BY PASSED PARAMETERS
C     IDENTIFYING THE IADS TYPE AND UNIT ELEMENT #; DELETES ALL FOUND
C     FOUND MESSAGES AND SHIFTS UP INTO DELETED SLOT REMAINING
C     MESSAGES ON HOLDING ARRAY. ALSO DELETE ALL MESSAGES GOING TO
C     THIS IADS TYPE AND UNIT #S.
C
C     CALLED BY - IADSPS
C     CALLS     - SHIFTU
C.....................................................................

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

      INTEGER TYPE,UNIT
```

```
C.....DECLARE LOCAL VARIABLES.....

      INTEGER FROMTYPE,FROMUNIT,II,LL,MESID,TOTYPE,TOUNIT
      REAL MESTIME

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG

C.....COMMON BLOCK.....
      INCLUDE 'HOLDM1 DEF'

      COMMON/DEBUG/DEBUG

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C.....LOCAL TO THIS SUBROUTINE.....
C     FROMTYPE - IADS TYPE # QUERIED MESSAGE IS COMING FROM
C     FROMUNIT - IADS UNIT # QUERIED MESSAGE IS COMING FROM
C     II - LOOP COUNTER
C     LL - LOOP COUNTER
C     MESID - HOLDS MESSAGE ID #
C     MESTIME - SENDING TIME OF QUERIED MESSAGE
C     TOTYPE - IADS TYPE # QUERIED MESSAGE IS GOING TO
C     TOUNIT - IADS UNIT # QUERIED MESSAGE IS GOING TO
C.....START PROGRAM.....

C     WRITE(*,*) 'DELMES'

C ... CHECK IF ANY MESSAGES MATCH THE TYPE AND UNIT NUMBER.....
C ... CONSTRUCT A LOOP THAT WILL ALLOW POINTING AND ENDING INDEX
C ... TO BE MODIFIED WHEN MESSAGES ARE TAKEN OFF ...

      II=1

   10 MESTIME=MESHOLD(II,2)
      FROMTYPE=NINT(MESHOLD(II,4))
      FROMUNIT=NINT(MESHOLD(II,5))
      TOTYPE=NINT(MESHOLD(II,6))
      TOUNIT=NINT(MESHOLD(II,7))
      MESID=NINT(MESHOLD(II,9))

      IF(DEBUG .GE. 3) THEN
        WRITE(16,*)'  MES. POINTER =',II,' LOOKING FOR TYPE,UNIT',
```

```
     &   TYPE,UNIT
         ENDIF

C ... IF TIME EQUALS ZERO THEN WE ARE AT THE END OF THE ARRAY ....

         IF(NINT(MESHOLD(II,1)) .EQ. 0) THEN
           IF(DEBUG .GE. 3) THEN
             WRITE(16,*)'   MESSAGE POINTER =',II,' MESHOLD(II,1)=0',
     &       ' AT END OF ARRAY'
           ENDIF
           GO TO 100
         ENDIF

C ... LOOK FOR MATCHING TYPE AND UNIT NUMBERS.....

         IF (DEBUG .GE. 3) THEN
           WRITE(16,*) '   MESSAGE QUERIED :',
     &   (MESHOLD(II,LL),LL=1,MAXMSGSIZE)
         ENDIF

         IF(FROMTYPE .EQ. TYPE .AND. FROMUNIT .EQ. UNIT   .OR.
     &    TOTYPE  .EQ. TYPE .AND.  TOUNIT  .EQ. UNIT) THEN
           IF(DEBUG .GE. 3) THEN
             WRITE(16,*) '   FOUND A MESSAGE THAT WILL BE DELETED',
     &       '; MSGID # =',MESID
           ENDIF
           IF (DEBUG .GE. 3) THEN
             WRITE(16,*) '   MESSAGE TO BE DELETED IS:',
     &       (MESHOLD(II,LL),LL=1,MAXMSGSIZE)
           ENDIF

C ... DELETE THIS MESSAGE BY SHIFTING ALL FOLLOWING MESSAGES UP ONE

           CALL SHIFTU(II)

         ENDIF

C... INCREMENT POINTER, IF > 200 THEN AT END OF ARRAY AND STOP LOOP ..

         II=II+1
         IF(II .GT. 200) THEN
           GO TO 100
         ENDIF

         GO TO 10

 100  CONTINUE

         RETURN
         END
```

```
------------------------------
      SUBROUTINE DLYMES(TYPE,UNIT,DELAY)

C.....LATEST CHANGE MADE 07/26/91 BFS.....
C     FINDS ALL MESSAGES IN HOLDING ARRAY WAITING TO BE SENT AND MATCHES
C     TO THE TYPE AND UNIT PASSED. MODIFYIES THE TIME THIS MESSAGE WILL
C     WILL BE SENT. THIS IS AS A RESULT OF THE UNIT BEING DAMAGED.
C
C     CALLED BY - IADSPS
C     CALLS     - NONE
C.....


      PARAMETER (IM=30,JM=25,KM=20)


      IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

      INTEGER TYPE,UNIT
      REAL DELAY
C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG

C.....COMMON BLOCK.....

      INCLUDE 'HOLDM1 DEF'

      COMMON/DEBUG/DEBUG

C.....DECLARE LOCAL VARIABLES.....

      INTEGER II,LL
      REAL OLDTIME

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C.....LOCAL TO THIS SUBROUTINE.....
C     II - LOOP COUNTER
C     OLDTIME - OROGINAL TIME MESSAGE WAS READY TO BE SENT
C.....START PROGRAM.....
```

```
C     WRITE(*,*) 'DLYMES'

C ... SEARCH FOR ANY MESSAGES THAT MATCH THE SENDING TYPE AND UNIT #
C ... IF ANY MATCH, MODIFY THE TIME WHEN THE MESSAGE WILL BE SEND

      DO 10 II=1,NEXTHOLDPOS-1

      IF(DEBUG .GE. 3) THEN
        WRITE(16,*)'   MES. POINTER =',II,' IADS TYPE =',TYPE,
     &   ' IADS UNIT #',UNIT
      ENDIF
      IF(DEBUG .GE. 3) THEN
        WRITE(16,*) '   MESSAGE QUERIED :',
     &   (MESHOLD(II,LL),LL=1,MAXMSGSIZE)
      ENDIF

      IF(TYPE .EQ. NINT(MESHOLD(II,4))  .AND.
     &   UNIT .EQ. NINT(MESHOLD(II,5))) THEN
        OLDTIME=MESHOLD(II,2)
        MESHOLD(II,2)=MESHOLD(II,1)+DELAY
        IF(DEBUG .EQ. 3) THEN
          WRITE(16,*) '   THIS MESSAGE WILL BE MODIFIED'
          WRITE(16,*) '   PREVIOUS TIME TO BE SENT=',OLDTIME
          WRITE(16,*) '    NOW NEW TIME TO BE SENT=',MESHOLD(II,2)
        ENDIF
        IF(DEBUG .GE. 3) THEN
          WRITE(16,*) '   MODIFIED MESSAGE :',
     &     (MESHOLD(II,LL),LL=1,MAXMSGSIZE)
        ENDIF
      ELSE
        IF(DEBUG .EQ. 3) THEN
          WRITE(16,*) ' THIS MESSAGE IS NOT TO BE DELAYED'
        ENDIF
      ENDIF
   10 CONTINUE

      RETURN
      END
-----------------------------
      SUBROUTINE EWDET

C.....LATEST CHANGE MADE 07/02/91 BFS.....
C     DETERMINES IF EW SITE DETECTS ANY PENETRATORS. FIRST INITIALIZES
C     THE NEWDET(*) AN ARRAY WHICH WILL HOLD THE PRESENT DETECTION
C     RESULTS. ALL SITES ARE QUERIED TO DETERMINE WHICH PENETRATORS
C     THEY NOW DETECT. FINALLY THE PRESENT DETECTION IS COMPARED TO
C     TO PREVIOUSLY DETERMINED STATE. IF NEW DETECTIONS OCCUR OR IF
C     PREVIOUS DETECTIONS ARE LOST, THIS IS NOTED.
C
C
```

```
C     CALLED BY - IADS
C     CALLS     - TERRA,CDELAY,EWMES
C.....................................................................

      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER IK,JJ,L,NEWDETECT(9,IM),CHANGE
      REAL DELAY,TDELAY
C.....DECLARE VARIABLES IN COMMON BLOCK.....

      REAL CD1,CD2,CD3,CD4,PSA,PSAMIN,RRAD,TIME,TTRACK,XT,XXINT,YT,ZT,
     &PS
      INTEGER DEBUG,I,IOP,ISECT,ISHOT,ITAR,IX,J,MSITE,NAC,NSITE,STATS

C.....COMMON BLOCK.....
      INCLUDE 'QIADS DEF'
      INCLUDE 'EW1 DEF'
      INCLUDE 'EWDATA DEF'

      COMMON/DEBUG/DEBUG
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     CD1(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM ENTERS OUTER
C                - LAUNCH ENVLOPE OF SAM SITE JM
C     CD2(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS IS
C                - THE ENVELOPE EXIT TIME. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE EXIT OF THE ENVELOPE INTO THE SITES
C                - MIN RANGE ZONE.
C     CD3(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
```

```
C              - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C              - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C              - THIS IS THE ENTRANCE INTO THE ENVELOPE FROM THE
C              - SITES MIN RANGE ZONE.
C     CD4(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C              - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C              - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C              - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C              - THIS IS THE EXIT OF THE PENETRATOR FROM THE ENVELOPE
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     I - PEN. LOOP COUNTER
C     IOP(10) - HOLDS FLAGS(1/0) TO SELECT PROGRAM RUNNING MODES
C     ISECT(IM,JM) - HOLDS A FLAG(1/0) FOR EACH PENETRATOR IM WHICH
C                - INDICATES WHETHER EACH SAM SITE JM CAN(=1) OR
C                - CANNOT(=0) FIRE AT THIS PENETRATOR
C     ISHOT(IM,JM)
C     ITAR(JM) - HOLDS PEN. # THAT SAM SITE JM IS LOCKED ONTO
C     IX(JM) - FLAG (1/0) USED TO CONTROL PRINTING OUTPUT DATA
C     J - SAM SITE LOOP COUNTER
C     MSITE(JM) - HOLDS THE SAM SITE TYPE (1-9) FOR SITE JM
C     NAC - MAX # OF PENETRATORS
C     NSITE - MAX # OF SAM SITES
C     PS(JM) - SURVIVAL PROBABILITY OF SITE JM
C     PSA(IM) - SURVIVAL PROBABILITY OF PENETRATOR IM
C     PSAMIN - MINIMUM SURVIVABILITY OF PENETRATOR;IF PSA(IM) IS BELOW
C          - THIS, THE PENETRATOR IS REMOVED FROM THE SIMULATION
C     RRAD(JM) - HOLDS TIMING DATA (IN SECONDS) FOR SAM SITE JM; USED
C              - IN DETERMINING WHEN TO FIRE,RE-FIRE,TRACK,ETC...
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C              - IF =0; SITE IS NOT RADIATING
C              - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C              - AND GETTING READY TO FIRE
C              - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C              - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C     TTRACK(JM)
C     XT(IM,2)
C     XXINT(JM)
C     YT(IM,2)
C     ZT(IM,2)
C.....LOCAL TO THIS SUBROUTINE.....
C  CHANGE - FLAG(1/O) =1 INDICATES THAT A CHANGE IN DETECTION OF PENS.
C         - (I.E. LOST OLD OR FOUND NEW DETECTION) HAS HAPPENED
C     IK - FLAG(1/0) USED BY SUBROUTINE JAMMER TO INDICATE IF(=1)
C        - THE PENETRATOR IS MASKED TO A SAM SITE BY JAMMING; BY
C        - SUBROUTINE TERRA TO INDICATE IF(=1) THE PENETRATOR IS MASKED
C        - BY THE TERRAIN FROM A SAM SITE; BY SUBROUTINE CHECKS TO
C        - INDICATE(=1) THAT THE PENETRATOR HAS BEEN DESTROYED; BY
C        - SUBROUTINE FUTLOK TO SEE IF(=1) THE ENGAGEMENT ANGLE OF THE
C        - PENETRATOR AND THE SAM IS WITHIN THE VALUE TO ALLOW FOR A
```

```
C       - LAUNCH AGAINST THIS PENETRATOR
C     JJ - LOOP COUNTER
C     L - LOOP COUNTER
C  NEWDETECT(9,IM) - TEMP ARRAY TO HOLD PRESENT DETECTION STATUS
C                 - USED TO COMPARE WITH OLD DETECTION STATUS AND
C                 - NOTE ANY CHANGES
C.....START PROGRAM.....

C     WRITE(*,*) 'EWDET'

      DO 10 JJ=1,NEWSITES
      DO 20 I=1,NAC
      NEWDETECT(JJ,I)=0
      UPEWTGT(JJ,I)=0
   20 CONTINUE
   10 CONTINUE

      DO 30 JJ=1,NEWSITES

C ... SKIP OVER THIS EW SITE IF IT IS NOT TURNED ON.....

      IF (DEBUG .GE. 3) THEN
        WRITE(16,*) 'EW SITE #, STATUS',JJ,STATS(EWID(JJ))
      ENDIF
      IF (STATS(EWID(JJ)) .EQ. 0) GO TO 30

C ... SKIP OVER THIS EW SITE IF IT IS DEAD.....

      IF (DEBUG .GE. 3) THEN
        WRITE(16,*) 'EW SITE #,PS,EWPSMIN',JJ,PS(EWID(JJ)),EWPSMIN
      ENDIF
      IF (PS(EWID(JJ)) .LT. EWPSMIN) GO TO 30

      CHANGE=0

      DO 70 I=1,NAC
      J=EWID(JJ)

C ... CHECK FOR TERRAIN MASKING OF PENETRATOR .....

      IF (PSA(I) .LT. PSAMIN) GO TO 65
      CALL TERRA(IK)
      IF (IK .NE. 0) GO TO 65
C.....POSSIBLE FUTURE MODS INCLUDING JAMMMING OF EW SITES .....
C     CALL JAMMER(IK)
C     IF (IK .NE. 0) ISHOT(I,J)=1
C     IF (IK .NE. 0) GO TO 65
C.....

C.....CHECK FOR PENETRATOR WITHIN EW DETECTION CONE.....
```

```
C.....NO PENETRATOR WITHIN DETECTION CONE.....
C.....IF ANY OF THESE TESTS PASS .....
      IF (DEBUG .GE. 3) THEN
        WRITE(16,*) 'TIME, WITH EW SITE #,PEN #',TIME,JJ,I
        WRITE(16,*) 'CD1-CD4',CD1(I,J),CD2(I,J),CD3(I,J),CD4(I,J)
      ENDIF
      IF (CD1(I,J) .GT. 20000.) GO TO 65
      IF (TIME .LT. CD1(I,J) .OR. TIME .GT. CD4(I,J)) GO TO 65
      IF (TIME .GT. CD2(I,J)  .AND.  TIME .LT. CD3(I,J)) GO TO 65

C ... TO GET TO NEXT LINE, PENETRATOR IS DETECTED .....

      NEWDETECT(JJ,I)=1

C ... TO GO TO 65 DIRECTLY MEANS PENETRATOR IS UNDETECTED .....

   65 CONTINUE

C ... TEST IF DETECTION STATUS OF PENETRATOR HAS CHANGED.....
      IF(NEWDETECT(JJ,I) .EQ. 1 .AND. EWDETECT(JJ,I) .EQ. 0) THEN
        EWDETECT(JJ,I)=1
        UPEWTGT(JJ,I)=1
        CHANGE=1
        IF (DEBUG .GE. 1) THEN
          WRITE(16,1001) TIME,JJ,I,XT(I,2),YT(I,2),ZT(I,2)
 1001   FORMAT(F5.0,' SEC EW SITE ',I2,' DETECTS PEN. ',I2,4X,F6.2,3X,
     $      F6.2,3X,F6.2)
        ENDIF
      ELSEIF(NEWDETECT(JJ,I) .EQ. 0 .AND. EWDETECT(JJ,I) .EQ. 1) THEN
        EWDETECT(JJ,I)=0
        UPEWTGT(JJ,I)=-1
        CHANGE=1
        IF (DEBUG .GE. 1) THEN
          WRITE(16,1002) TIME,JJ,I,XT(I,2),YT(I,2),ZT(I,2)
 1002   FORMAT(F5.0,' SEC EW SITE ',I2,' LOST DETECTION PEN. ',I2,4X,
     $      F6.2,3X,F6.2,3X,F6.2)
        ENDIF
      ENDIF


   70 CONTINUE

C ... IF ANY CHANGE IN DETECTION STATUS HAS OCCURED FOR THIS EW SITE
C ... SEND A MESSAGE .....

      IF(CHANGE .EQ. 1) THEN
        IF (DEBUG .GE. 3) THEN
          WRITE(16,*) ' INPUT DELAY PARAMETERS:'
          WRITE(16,*) EWPROCDLY(JJ),EWDAMDLY(JJ),EWDAMRAT(JJ),PS(J)
        ENDIF
```

```
      CALL CDELAY(EWPROCDLY(JJ),EWDAMDLY(JJ),EWDAMRAT(JJ),
     &  PS(J),DELAY)
        IF (DEBUG .GE. 3) THEN
          WRITE(16,*) ' CHANGE IN DETECTION STATUS FOR EW SITE #',JJ
          WRITE(16,*) '(-1= LOST, 0= NO CHANGE, 1= NEW DETECTION)'
          WRITE(16,*) ' (1-NAC) UPDATE DETECTION ARRAY :',
     &    (UPEWTGT(JJ,L),L=1,NAC)
          WRITE(16,*) ' COMPUTED DELAY=',DELAY,
     &    '  TIME WHEN MESSAGE WILL BE SENT=',TIME+DELAY
        ENDIF
        IF (DEBUG .GE. 1) THEN
          WRITE(16,1003) TIME,JJ,TIME+DELAY
1003      FORMAT(F5.0,' SEC EW SITE ',I2,' WILL SEND MESSAGE',
     &    ' AT TIME ',F5.0)
        ENDIF
        TDELAY=TIME+DELAY
        CALL EWMES(JJ,TDELAY)
      ENDIF

   30 CONTINUE

C.....

      RETURN
      END
---------------------------------
      SUBROUTINE EWDET2(JJ,UPDATE)

C.....LATEST CHANGE MADE 07/29/91 BFS.....
C     SIMPLIFIED VERSION OF EWDET SURBROUTINE, USED WHEN AN EW SITE HAS
C     BEEN DESTROYED AND AN UPDATED DETECTION STATUS IS NEEDED. THIS
C     WILL DETERMINE ALL THE PENETRATORS THAT THE PASSED SITE DETECTS
C
C     CALLED BY - IADSPS
C     CALLS     - TERRA                                              .
C.....

      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

      INTEGER JJ,UPDATE(IM)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER II,IK,J1

C.....DECLARE VARIABLES IN COMMON BLOCK.....
```

```
      REAL CD1,CD2,CD3,CD4,PSA,PSAMIN,TIME
      INTEGER DEBUG,NAC

C.....COMMON BLOCK.....

      INCLUDE 'QIADS DEF'

      COMMON/DEBUG/DEBUG
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     CD1(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM ENTERS OUTER
C                - LAUNCH ENVLOPE OF SAM SITE JM
C     CD2(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS IS
C                - THE ENVELOPE EXIT TIME. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE EXIT OF THE ENVELOPE INTO THE SITES
C                - MIN RANGE ZONE.
C     CD3(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C                - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE ENTRANCE INTO THE ENVELOPE FROM THE
C                - SITES MIN RANGE ZONE.
C     CD4(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C                - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE EXIT OF THE PENETRATOR FROM THE ENVELOPE
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     IOP(10) - HOLDS FLAGS(1/0) TO SELECT PROGRAM RUNNING MODES
C     ISECT(IM,JM) - HOLDS A FLAG(1/0) FOR EACH PENETRATOR IM WHICH
C                  - INDICATES WHETHER EACH SAM SITE JM CAN(=1) OR
C                  - CANNOT(=0) FIRE AT THIS PENETRATOR
C     ISHOT(IM,JM)
C     ITAR(JM) - HOLDS PEN. # THAT SAM SITE JM IS LOCKED ONTO
C     IX(JM) - FLAG (1/0) USED TO CONTROL PRINTING OUTPUT DATA
C     MSITE(JM) - HOLDS THE SAM SITE TYPE (1-9) FOR SITE JM
```

```
C     NAC - MAX # OF PENETRATORS
C     NSITE - MAX # OF SAM SITES
C     PS(JM) - SURVIVAL PROBABILITY OF SITE JM
C     PSA(IM) - SURVIVAL PROBABILITY OF PENETRATOR IM
C     PSAMIN - MINIMUM SURVIVABILITY OF PENETRATOR;IF PSA(IM) IS BELOW
C           - THIS, THE PENETRATOR IS REMOVED FROM THE SIMULATION
C     RRAD(JM) - HOLDS TIMING DATA (IN SECONDS) FOR SAM SITE JM; USED
C            - IN DETERMINING WHEN TO FIRE,RE-FIRE,TRACK,ETC...
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C              - IF =0; SITE IS NOT RADIATING
C              - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C              - AND GETTING READY TO FIRE
C              - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C              - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C     TTRACK(JM)
C     XT(IM,2)
C     XXINT(JM)
C     YT(IM,2)
C     ZT(IM,2)
C.....LOCAL TO THIS SUBROUTINE.....
C  CHANGE - FLAG(1/O) =1 INDICATES THAT A CHANGE IN DETECTION OF PENS.
C        - (I.E. LOST OLD OR FOUND NEW DETECTION) HAS HAPPENED
C     IK - FLAG(1/0) USED BY SUBROUTINE JAMMER TO INDICATE IF(=1)
C       - THE PENETRATOR IS MASKED TO A SAM SITE BY JAMMING; BY
C       - SUBROUTINE TERRA TO INDICATE IF(=1) THE PENETRATOR IS MASKED
C       - BY THE TERRAIN FROM A SAM SITE; BY SUBROUTINE CHECKS TO
C       - INDICATE(=1) THAT THE PENETRATOR HAS BEEN DESTROYED; BY
C       - SUBROUTINE FUTLOK TO SEE IF(=1) THE ENGAGEMENT ANGLE OF THE
C       - PENETRATOR AND THE SAM IS WITHIN THE VALUE TO ALLOW FOR A
C       - LAUNCH AGAINST THIS PENETRATOR
C     II - LOOP COUNTER
C     J1 - SAM UNIT # FOR THIS EW SITE
C     JJ - LOOP COUNTER
C     L - LOOP COUNTER
C  NEWDETECT(9,IM) - TEMP ARRAY TO HOLD PRESENT DETECTION STATUS
C                - USED TO COMPARE WITH OLD DETECTION STATUS AND
C                - NOTE ANY CHANGES
C.....START PROGRAM.....

C     WRITE(*,*) 'EWDET2'

      DO 20 II=1,NAC
      UPDATE(II)=0
   20 CONTINUE


      DO 70 II=1,NAC
      J1=EWID(JJ)
```

```
C ... CHECK FOR TERRAIN MASKING OF PENETRATOR .....

      IF (PSA(II) .LT. PSAMIN) GO TO 65
      CALL TERRA(IK)
      IF (IK .NE. 0) GO TO 65

C.....CHECK FOR PENETRATOR WITHIN EW DETECTION CONE.....

C.....NO PENETRATOR WITHIN DETECTION CONE.....
C.....IF ANY OF THESE TESTS PASS .....
      IF (DEBUG .GE. 3) THEN
        WRITE(16,*) 'TIME, WITH EW SITE #,PEN #',TIME,J1,II
        WRITE(16,*) 'CD1-CD4',CD1(II,J1),CD2(II,J1),CD3(II,J1),
     &   CD4(II,J1)
      ENDIF
      IF (CD1(II,J1) .GT. 20000.) GO TO 65
      IF (TIME .LT. CD1(II,J1) .OR. TIME .GT. CD4(II,J1)) GO TO 65
      IF (TIME .GT. CD2(II,J1)  .AND.  TIME .LT. CD3(II,J1)) GO TO 65

C ... TO GET TO NEXT LINE, PENETRATOR IS DETECTED .....

      UPDATE(II)=1

C ... TO GO TO 65 DIRECTLY MEANS PENETRATOR IS UNDETECTED .....

  65  CONTINUE

  70 CONTINUE


      RETURN
      END
-----------------------------------
      SUBROUTINE EWMES(JJ,TDELAY)

C.....LATEST CHANGE MADE 07/23/91 BFS.....
C     RECEIVES UPDATED EW SITE DETECTION MESSAGE DATA, ORGANIZES THE
C     DATA TO BE SENT, IDENTIFIES THE IADS ELEMENTS TO SEND MESSAGES
C     TO AND CALLS A SUB WHICH PUTS THE MESSAGE IN A HOLDING ARRAY
C
C
C     CALLED BY - EWDET
C     CALLS     - HOLDMES
C.....

      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....
```

```
      INTEGER JJ
      REAL TDELAY
C.....DECLARE VARIABLES IN COMMON BLOCK.....

      REAL TIME,PS
      INTEGER DEBUG,IOP,J,NAC

C.....COMMON BLOCK.....

      INCLUDE 'QIADS DEF'
      INCLUDE 'EW1 DEF'
      INCLUDE 'HOLDM1 DEF'
      INCLUDE 'CON1 DEF'
      INCLUDE 'MESID DEF'

      COMMON/DEBUG/DEBUG
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C.....DECLARE LOCAL VARIABLES.....
      INTEGER L,LL,POS
      REAL MSG(MAXMSGSIZE)
C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     IOP(10) - HOLDS FLAGS(1/0) TO SELECT PROGRAM RUNNING MODES
C     J - SAM SITE LOOP COUNTER
C     NAC - MAX # OF PENETRATORS
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C.....LOCAL TO THIS SUBROUTINE.....
C   L - LOOP COUNTER
C   LL - LOOP COUNTER
C   MSG(MAXMSGSIZE) - HOLDS MESSAGE PARTICULARS
C   POS - INPUT INDEX OF MESSAGE IN HOLDING ARRAY
C.....START PROGRAM.....

C     WRITE(*,*) 'EWMES'

      POS=EWMESPOS(JJ)
```

```
      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) 'EW MESSAGE POINTER IS NOW =',EWMESPOS(JJ)
        WRITE(16,*) 'MESSAGE ID # IS NOW =',MSGID
      ENDIF

      IF(POS .GT. MAXEWMES) THEN
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(*,*) 'EW SITE #',JJ,' NEEDS TO SEND A MESSAGE AND',
     &   ' EW MESSAGE ARRAY IS FULL'
        WRITE(16,*) 'EW SITE #',JJ,' NEEDS TO SEND A MESSAGE AND',
     &   ' EW MESSAGE ARRAY IS FULL'
        WRITE(*,*) 'PRESENT VALUE OF MAXEWMES PARAMETER IS',POS-1
        WRITE(*,*) 'INCREASE MAXEWMES PARAMETER IN EWMES1 DEF AND',
     &   ' RECOMPILE PROGRAM MODULES'
        WRITE(16,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
      STOP
      ENDIF

C ... CONSTRUCT THE MESSAGE .....

      EWDETMES(JJ,1,POS)=TIME
      EWDETMES(JJ,2,POS)=TDELAY
      EWDETMES(JJ,3,POS)=PS(EWID(JJ))
      EWDETMES(JJ,4,POS)=1.0
      EWDETMES(JJ,5,POS)=REAL(JJ)
      EWDETMES(JJ,8,POS)=0.0
      EWDETMES(JJ,9,POS)=REAL(MSGID)
      EWDETMES(JJ,10,POS)=1.0
      EWDETMES(JJ,11,POS)=REAL(JJ)
      EWDETMES(JJ,12,POS)=3.0
      EWDETMES(JJ,13,POS)=1.0

      MSG(1)=TIME
      MSG(2)=TDELAY
      MSG(3)=PS(EWID(JJ))
      MSG(4)=1.0
      MSG(5)=REAL(JJ)
      MSG(8)=0.0
      MSG(9)=REAL(MSGID)
      MSG(10)=1.0
      MSG(11)=REAL(JJ)
      MSG(12)=3.0
      MSG(13)=1.0

      DO 10 L=1,NAC
      EWDETDAT(JJ,L,POS)=UPEWTGT(JJ,L)
   10 CONTINUE

      IF (DEBUG .GE. 3) THEN
```

```
      WRITE(16,*) 'MESSAGE DATA FROM EW SITE #',
     &              NINT(EWDETMES(JJ,5,POS)), ' MESSAGE #',POS
      WRITE(16,*)'EWDETMES(JJ,L);L=1,?',
     &  (EWDETMES(JJ,L,POS),L=1,MAXMSGSIZE)
      WRITE(16,*) 'EWDETDAT(JJ,L,POS);L=1,NAC',
     &            (EWDETDAT(JJ,L,POS),L=1,NAC)
     ENDIF

C.....DETERMINE WHERE MESSAGE IS GOING VIA CONNECT ARRAY .....
C.... LOOK FIRST FOR A MATCH OF EW TYPE AND CORRESPONDING UNIT # ..

     DO 20 L=1,NUMCON
     IF(CONNECT(L,1) .EQ. 1  .AND. CONNECT(L,2) .EQ. JJ) THEN
       IF(CONNECT(L,3) .EQ. 2) THEN
         IF (DEBUG .GE. 2) THEN
           WRITE(16,*) 'EW SITE #',JJ,' WILL SEND MESSAGE TO EW',
     &       ' RPT STA #',CONNECT(L,4),' MSGID=',MSGID
         ENDIF
         IF (DEBUG .GE. 3) THEN
           WRITE(16,*) 'MESSAGE TO BE SEND IS:',
     &       (MSG(LL),LL=1,MAXMSGSIZE)
         ENDIF
       ELSEIF(CONNECT(L,3) .EQ. 3) THEN
         IF (DEBUG .GE. 2) THEN
           WRITE(16,*) 'EW SITE #',JJ,' WILL SEND MESSAGE TO',
     &       ' ADWOC CENTER; MSGID=',MSGID
         ENDIF
       ELSEIF(CONNECT(L,3) .EQ. 4) THEN
         IF (DEBUG .GE. 2) THEN
           WRITE(16,*) 'EW SITE #',JJ,' WILL SEND MES. TO BAT. #',
     &       CONNECT(L,4),'MSGID=',MSGID
         ENDIF
       ELSEIF(CONNECT(L,3) .EQ. 5) THEN
         IF (DEBUG .GE. 2) THEN
           WRITE(16,*) 'EW SITE #',JJ,' WILL SEND MES. TO SAM SITE',
     &       ' #',CONNECT(L,4),'MSGID=',MSGID
         ENDIF
       ENDIF
       MSG(6)=REAL(CONNECT(L,3))
       MSG(7)=REAL(CONNECT(L,4))
       EWDETMES(JJ,6,POS)=MSG(6)
       EWDETMES(JJ,7,POS)=MSG(7)
       CALL HOLDMES(MSG)
     ENDIF

  20 CONTINUE

C... INCREMENT POSITON INDEX AND ID # FOR NEXT EW MESSAGE .....

     MSGID=MSGID+1
```

```
      EWMESPOS(JJ) = POS + 1

      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) 'EW SITE MESSAGE POINTER INCREMENTED TO =',
     &    EWMESPOS(JJ),' FOR NEXT MESSAGE'
        WRITE(16,*) 'NEXT MESSAGE ID # =',MSGID,
     &    ' FOR NEXT MESSAGE'
      ENDIF


      RETURN
      END
-----------------------------------
      SUBROUTINE FIRE

C.....LATEST CHANGE MADE 6/28/91 BFS.....
C     COPY OF SHOT FROM NON-IADS MODE
C
C     CALLED BY - IADS
C     CALLS     - FIRE0,FIRE1,FIRE2,FIRE3
C.....

      PARAMETER (IM = 30,JM = 25,KM = 20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      REAL PS,PSSMIN,TIME
      INTEGER DEBUG,ITAR,J,NSITE,NOSHOT,NSHOT,STATS

C.....COMMON BLOCK.....
      INCLUDE 'QIADS DEF'

      COMMON/DEBUG/DEBUG
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/RCTIME/TRACK(9),TTRACK(JM),NUMSHT(JM),NOSHOT(JM),NSHOT(JM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
```

```
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     ITAR(JM) - HOLDS PEN. # THAT SAM SITE JM IS LOCKED ONTO
C     J - SAM SITE LOOP COUNTER
C     NSITE - MAX # OF SAM SITES
C     NOSHOT(JM) - MAX # OF SALVOS AVAILABLE TO SAM SITE (JM)
C     NSHOT(JM) - RUNNING TOTAL # OF SALVOS SAM SITE JM HAS FIRED
C     PS(JM) - SURVIVAL PROBABILITY OF SAM SITE JM
C     PSSMIN - MIN SURVIVAL PROBABILITY THAT THE SAM SITE MUST MAINTAIN
C           - IN ORDER TO STAY IN THE SIMULATION
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C               - IF =0; SITE IS NOT RADIATING
C               - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C               - AND GETTING READY TO FIRE
C               - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C               - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C.....START PROGRAM.....

C     WRITE(*,*) 'FIRE'

      DO 10 J=1,NSITE

C ... IF THIS IS AN EW SITE ,SKIP OVER IT .....
      IF(EWSITE(J) .EQ. 1) GOTO 10

      IF (PSSMIN .LE. PS(J)  .AND. NOSHOT(J) .GE. NSHOT(J)) GO TO 20

      IF (STATS(J) .NE. 0) THEN

        IF (NSHOT(J) .GT. NOSHOT(J)) THEN
          IF (DEBUG .GE. 1) THEN
            WRITE(16,1000) TIME,J
1001        FORMAT(F5.0,' SEC SITE ',I2,' STOPS RADIATING PERMANENTLY:',
     $      ' ALL SALVOS SHOT ')
          ENDIF
          WRITE(17,2000) 54,1
2001        FORMAT(I2,I2)
          WRITE(17,3000) TIME,0,J,0,0,0.0
3001        FORMAT(F5.0,I2,I2,I3,I3,F9.4)
        ENDIF

        IF(PS(J) .LT. PSSMIN) THEN
          IF (DEBUG .GE. 1) THEN
            WRITE(16,1000) TIME,J
1000        FORMAT(F5.0,' SEC SITE ',I2,' STOPS RADIATING PERMANENTLY:',
     $      'SITE DESTROYED')
          ENDIF
          WRITE(17,2000) 55,1
```

```
2000       FORMAT(I2,I2)
          WRITE(17,3000) TIME,0,J,0,0,0.0
3000       FORMAT(F5.0,I2,I2,I3,I3,F9.4)
        ENDIF
      ENDIF

      STATS(J)=0
      GO TO 10


   20 IF (STATS(J) .EQ. 0) THEN
          IF (ITAR(J) .NE. 0) ITAR(J)=0
          CALL FIRE0
          GO TO 10
        ELSEIF (STATS(J) .EQ. 1) THEN
          CALL FIRE1
          GO TO 10
        ELSEIF (STATS(J) .EQ. 2) THEN
          CALL FIRE2
          GO TO 10
        ELSEIF (STATS(J) .EQ. 3) THEN
          CALL FIRE3
      ENDIF

   10 CONTINUE


      RETURN
      END
-----------------------------------
      SUBROUTINE FIRE0

C.....LATEST CHANGE MADE  7/03/91 BFS.....
C     COPY OF SHOT0 FROM NON-IADS MODE
C
C     CALLED BY - FIRE
C     CALLS     - TERRA,JAMMER,FUTLOK,NTRCPT
C.....

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      REAL TF
      INTEGER IJ,IK,JJ,K,MFLG,TOT

C.....DECLARE VARIABLES IN COMMON BLOCK.....
```

```
      REAL CD1,CD2,CD3,CD4,PSA,PSAMIN,RRAD,TIME,TTRACK,XT,XXINT,YT,ZT
      INTEGER DEBUG,I,IOP,ISECT,ISHOT,ITAR,IX,J,MSITE,NAC,NSITE,STATS

C.....COMMON BLOCK.....

       INCLUDE 'KNOWN1 DEF'

      COMMON/CRUISE/IOP(10),TIMEL,RRAD(JM),IT(JM)
      COMMON/DEBUG/DEBUG
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/RCTIME/TRACK(9),TTRACK(JM),NUMSHT(JM),NOSHOT(JM),NSHOT(JM)
      COMMON/SHOTA/XXINT(JM),IX(JM),MASKTIM(JM),MWAITIM(JM),OUTIM(JM),
     $OWAITIM(JM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     CD1(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM ENTERS OUTER
C                - LAUNCH ENVLOPE OF SAM SITE JM
C     CD2(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS IS
C                - THE ENVELOPE EXIT TIME. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE EXIT OF THE ENVELOPE INTO THE SITES
C                - MIN RANGE ZONE.
C     CD3(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C                - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE ENTRANCE INTO THE ENVELOPE FROM THE
C                - SITES MIN RANGE ZONE.
C     CD4(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C                - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE EXIT OF THE PENETRATOR FROM THE ENVELOPE
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     I - PEN. LOOP COUNTER
C     IOP(10) - HOLDS FLAGS(1/0) TO SELECT PROGRAM RUNNING MODES
```

```
C     ISECT(IM,JM) - HOLDS A FLAG(1/0) FOR EACH PENETRATOR IM WHICH
C                  - INDICATES WHETHER EACH SAM SITE JM CAN(=1) OR
C                  - CANNOT(=0) FIRE AT THIS PENETRATOR
C     ISHOT(IM,JM)
C     ITAR(JM) - HOLDS PEN. # THAT SAM SITE JM IS LOCKED ONTO
C     IX(JM) - FLAG (1/0) USED TO CONTROL PRINTING OUTPUT DATA
C     J - SAM SITE LOOP COUNTER
C     MSITE(JM) - HOLDS THE SAM SITE TYPE (1-9) FOR SITE JM
C     NAC - MAX # OF PENETRATORS
C     NSITE - MAX # OF SAM SITES
C     PSA(IM) - SURVIVAL PROBABILITY OF PENETRATOR IM
C     PSAMIN - MINIMUM SURVIVABILITY OF PENETRATOR;IF PSA(IM) IS BELOW
C           - THIS, THE PENETRATOR IS REMOVED FROM THE SIMULATION
C     RRAD(JM) - HOLDS TIMING DATA (IN SECONDS) FOR SAM SITE JM; USED
C              - IN DETERMINING WHEN TO FIRE,RE-FIRE,TRACK,ETC...
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C               - IF =0; SITE IS NOT RADIATING
C               - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C               - AND GETTING READY TO FIRE
C               - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C               - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C     TTRACK(JM)
C     XT(IM,2)
C     XXINT(JM)
C     YT(IM,2)
C     ZT(IM,2)
C.....LOCAL TO THIS SUBROUTINE.....
C     IJ
C     IK - FLAG(1/0) USED BY SUBROUTINE JAMMER TO INDICATE IF(=1)
C        - THE PENETRATOR IS MASKED TO A SAM SITE BY JAMMING; BY
C        - SUBROUTINE TERRA TO INDICATE IF(=1) THE PENETRATOR IS MASKED
C        - BY THE TERRAIN FROM A SAM SITE; BY SUBROUTINE CHECKS TO
C        - INDICATE(=1) THAT THE PENETRATOR HAS BEEN DESTROYED; BY
C        - SUBROUTINE FUTLOK TO SEE IF(=1) THE ENGAGEMENT ANGLE OF THE
C        - PENETRATOR AND THE SAM IS WITHIN THE VALUE TO ALLOW FOR A
C        - LAUNCH AGAINST THIS PENETRATOR
C     JJ - LOOP COUNTER
C     K
C     MFLG
C     TOT - TOTAL NUMBER OF OTHER SAM SITES THAT ARE ENGAGING THE
C         - THE PENETRATOR UNDER CONSIDERATION
C     TF
C.....START PROGRAM.....

C     WRITE(*,*) 'FIRE0'

C.....SEARCH FOR TARGET INTERCEPTING CONE.....
C.....
```

```
      MFLG=0
      IX(J)=0

C.....PRINT FLAG FALSE.....

      DO 70 I=1,NAC
C....IF YOU HAVE NOT YET DETECTED THE TGT WITH YOUR EW SITES SKIP OVER.
      IF(DETALL(I) .EQ. 0) GO TO 70
C.....
      IF (ISECT(I,J) .EQ. 0  .OR. PSA(I) .LT. PSAMIN) GO TO 70
      CALL TERRA(IK)
      IF (IK .NE. 0) GO TO 70
      CALL JAMMER(IK)
      IF (IK .NE. 0) ISHOT(I,J)=1
      IF (IK .NE. 0) GO TO 70


C.....CHECK FOR PENETRATOR WITHIN SAM CONE.....

C.....NO INTERSECTION WITH CONE.....

      IF (CD1(I,J) .GT. 20000.) GO TO 70
      IF (TIME .LT. CD1(I,J)-TTRACK(J)  .OR. TIME .GT. CD4(I,J)-
     $TTRACK(J)) GO TO 70
      IF (TIME+TTRACK(J) .LT. CD2(I,J)) GO TO 80
      IF (CD3(I,J) .GT. 20000.) GO TO 70
      IF (TIME+TTRACK(J) .GT. CD2(I,J)  .AND.  TIME+TTRACK(J) .LT.
     $CD3(I,J)) GO TO 70

C.....CHECK FOR MINIMUM ASPECT ANGLE IF SAM IS AN IR MISSILE.....

   80 CALL FUTLOK(IK)
      IF (IK .NE. 0) GO TO 70

C.....  FOR THE ONE SITE VERSUS ONE TARGET DOCTRINE  .....
C.....IF ANY OTHER SITE IS ENGAGING THIS TARGET, YOU CAN'T.....

      IF (IOP(3) .EQ. 1) THEN
        DO 85 JJ=1,NSITE
C        WRITE(6,*) 'JJ,J,I,ITAR(JJ)', JJ,J,I,ITAR(JJ)
        IF (J .NE. JJ  .AND.  ITAR(JJ) .EQ. I) GO TO 70
   85   CONTINUE
      ENDIF


C.....  FOR THE MAX OF TWO SITES VERSUS ONE TARGET DOCTRINE  .....
C....IF MORE THAN ONE OTHER SITE IS ENGAGING THIS TARGET, YOU CAN'T..

      IF (IOP(3) .EQ. 2) THEN
        TOT=0
```

```
      DO 87 JJ=1,NSITE
      IF (J .NE. JJ  .AND.  ITAR(JJ) .EQ. I) TOT=TOT+1
   87   CONTINUE
     IF(TOT .GT. 1) GO TO 70
     ENDIF

C.....TO GET TO HERE MEANS THAT YOU HAVE AT LEAST ONE VALID TARGET.....
C.....MFLG >1 MEANS THAT LOCKON TO MORE THAN 1 TARGET IS POSSIBLE....

   90 MFLG=MFLG+1
C     IF (MFLG .GT. 1) GO TO 100
      IF (IOP(3) .EQ. 5  .AND.  MFLG .GT. 1) GO TO 100
      IF (IOP(3) .EQ. 3  .AND.  MFLG .GT. 1) GO TO 70
      IF (IOP(3) .EQ. 4  .AND.  MFLG .GT. 1) GO TO 70
      ITAR(J)=I
      CALL NTRCPT(TF)
      XXINT(J)=TF
      GO TO 70

C.............WITH LOCKON TO MORE THAN 1 TARGET POSSIBLE,.............
C..............................PICK THE ONE WITH THE LARGER Ps.............

  100 IJ=ITAR(J)
      IF (PSA(I) .LT. PSA(IJ)) GO TO 70
      CALL NTRCPT(TF)
      IF (TF .GT. XXINT(J)) GO TO 70
      ITAR(J)=I
      XXINT(J)=TF

   70 CONTINUE

C.....


      IF (MFLG .EQ. 0) GO TO 999


C.....1 MEANS SITE J IS ON LOCKON PHASE.....

      STATS(J)=1

        IF (DEBUG .GE. 1) THEN
          WRITE(16,1000) TIME,J
 1000       FORMAT(F5.0,' SEC SITE ',I2,' STARTS RADIATING')
        ENDIF
        WRITE(17,2000) 53,1
 2000    FORMAT(I2,I2)
        WRITE(17,3000) TIME,0,J,0,0,0.0
 3000    FORMAT(F5.0,I2,I2,I3,I3,F9.4)
```

```
      RRAD(J)=0.0
      I=ITAR(J)

      DO 110 K=1,NAC
      ISHOT(K,J)=0
  110 CONTINUE

      IF (DEBUG .GE. 1) THEN
        WRITE(16,1001) TIME,J,I,XT(I,2),YT(I,2),ZT(I,2)
 1001   FORMAT(F5.0,' SEC SITE ',I2,' LOCKS ON PEN. ',I2,4X,F8.2,3X,
     $  F8.2,3X,F8.2)
      ENDIF
C.....WRITE EVENT AND DATA TO DOC FILE.....

      WRITE(17,2001) 24,1
 2001 FORMAT(I2,I2)
      WRITE(17,3001) TIME,I,J,0,0,0.0
 3001 FORMAT(F5.0,I2,I2,I3,I3,F9.4)

  999 CONTINUE

      RETURN
      END
--------------------------------
      SUBROUTINE FIRE1

C.....LATEST CHANGE MADE  6/28/91 BFS.....
C     COPY OF SHOT1 FROM NON-IADS MODE
C
C     CALLED BY - FIRE
C     CALLS     - FIRE4,NTRCPT
C.....

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      REAL TF

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      REAL DT,RRAD,TIME,TTRACK,XT,XXINT,YT,ZT
      INTEGER DEBUG,I,ITAR,J,NSHOT,STATS

C.....COMMON BLOCK.....

      COMMON/CRUISE/IOP(10),TIMEL,RRAD(JM),IT(JM)
      COMMON/DEBUG/DEBUG
```

```
      COMMON/RCTIME/TRACK(9),TTRACK(JM),NUMSHT(JM),NOSHOT(JM),NSHOT(JM)
      COMMON/SHOTA/XXINT(JM),IX(JM),MASKTIM(JM),MWAITIM(JM),OUTIM(JM),
     $OWAITIM(JM)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                                  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     DT - TACOPS TIME INCREMENT (1 SECOND)
C     I - PEN. LOOP COUNTER
C     ITAR(JM) - HOLDS PEN. # THAT SAM SITE JM IS LOCKED ONTO
C     J - SAM SITE LOOP COUNTER
C     NSHOT(JM) - RUNNING TOTAL # OF SALVOS SAM SITE JM HAS FIRED
C     RRAD(JM) - HOLDS TIMING DATA (IN SECONDS) FOR SAM SITE JM; USED
C              - IN DETERMINING WHEN TO FIRE,RE-FIRE,TRACK,ETC...
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C               - IF =0; SITE IS NOT RADIATING
C               - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C               - AND GETTING READY TO FIRE
C               - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C               - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C     TTRACK(JM)
C     XT(IM,2)
C     XXINT(JM)
C     YT(IM,2)
C     ZT(IM,2)
C.....LOCAL TO THIS SUBROUTINE.....
C     TF
C.....START PROGRAM.....

C     WRITE(*,*) 'FIRE1'

C.....SHOOT SAM UNLESS TARGET TRACK IS LOST.....

      I=ITAR(J)
      RRAD(J)=RRAD(J)+DT

C.....TOO EARLY TO SHOOT.....

      IF (RRAD(J) .LT. TTRACK(J)) GO TO 10
      GO TO 20
```

```
  10 CALL FIRE4
     GO TO 99

  20 CONTINUE

C.....2 MEANS THERE'S A SAM ENROUTE.....

      STATS(J)=2
      CALL NTRCPT(TF)
      XXINT(J)=TF
      NSHOT(J)=NSHOT(J)+1
      IF (DEBUG .GE. 1) THEN
        WRITE(16,1001) TIME,J,I,NSHOT(J),XT(I,2),YT(I,2),ZT(I,2)
 1001   FORMAT(F5.0,' SEC SITE ',I2,' FIRES ON PEN. ',I2,' SALVO ', I3,
     $  4X,F8.2,1X,F8.2,1X,F8.2)
      ENDIF
C.....WRITE EVENT AND DATA TO DOC FILE.....

      WRITE(17,2001) 25,1
 2001 FORMAT(I2,I2)
      WRITE(17,3001) TIME,I,J,NSHOT(J),0,0.0
 3001 FORMAT(F5.0,I2,I2,I3,I3,F9.4)


  99 RETURN
     END
---------------------------------
      SUBROUTINE FIRE2

C.....LATEST CHANGE MADE  7/01/91 BFS.....
C     COPY OF SHOT2 FROM NON-IADS MODE
C
C     CALLED BY - FIRE
C     CALLS     - TERRA,ASSESS
C.....

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)
C.....DECLARE LOCAL VARIABLES.....

      INTEGER IK

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      REAL CD1,CD2,CD3,CD4,DT,GDT,MASKTIM,MWAITIM,OUTIM,OWAITIM,RRAD,
     $TIME,TTRACK,XT,XXINT,YT,ZT
      INTEGER DEBUG,I,IOP,ITAR,J,STATS

C.....COMMON BLOCK.....
```

```
      COMMON/CRUISE/IOP(10),TIMEL,RRAD(JM),IT(JM)
      COMMON/DEBUG/DEBUG
      COMMON/RCTIME/TRACK(9),TTRACK(JM),NUMSHT(JM),NOSHOT(JM),NSHOT(JM)
      COMMON/SHOTA/XXINT(JM),IX(JM),MASKTIM(JM),MWAITIM(JM),OUTIM(JM),
     $OWAITIM(JM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....IN COMMON BLOCK.....
C     CD1(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM ENTERS OUTER
C                - LAUNCH ENVLOPE OF SAM SITE JM
C     CD2(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS IS
C                - THE ENVELOPE EXIT TIME. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE EXIT OF THE ENVELOPE INTO THE SITES
C                - MIN RANGE ZONE.
C     CD3(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C                - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE ENTRANCE INTO THE ENVELOPE FROM THE
C                - SITES MIN RANGE ZONE.
C     CD4(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C                - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE EXIT OF THE PENETRATOR FROM THE ENVELOPE
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     DT - TACOPS TIME INCREMENT (1 SECOND)
C     GDT(JM) - KILL ASSES DELAY TIME FOR SAM SITE JM
C     I - PEN. LOOP COUNTER
C     IOP(10) - HOLDS FLAGS(1/0) TO SELECT PROGRAM RUNNING MODES
C     ITAR(JM) - HOLDS PEN. # THAT SAM SITE JM IS LOCKED ONTO
C     J - SAM SITE LOOP COUNTER
C     MASKTIM(J) - TIME (IN SECONDS) WHEN FIRED ON TARGET IS TERRAIN
C                - MASKED
C     MWAITIM(J) - MAXIMUM WAIT TIME (IN SECONDS) AFTER THE TARGET IS
C                - TERRAIN MASKED BEFORE THE SITE LOSES TRACK AND THE
C                - OUTBOUND SAM IS LOST
C     OUTIM(J) - TIME (IN SECONDS) WHEN FIRED ON TARGET IS OUTSIDE OF
```

```
C            - THE SITE LAUNCH ENVELOPE
C     OWAITIM(J) - MAXIMUM WAIT TIME (IN SECONDS) AFTER THE TARGET IS
C              - OUTSIDE OF THE SITE LAUNCH ENVELOPE BEFORE THE SITE
C              - LOSES TRACK AND THE OUTBOUND SAM IS LOST
C     RRAD(JM) - HOLDS TIMING DATA (IN SECONDS) FOR SAM SITE JM; USED
C              - IN DETERMINING WHEN TO FIRE,RE-FIRE,TRACK,ETC...
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C              - IF =0; SITE IS NOT RADIATING
C              - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C              - AND GETTING READY TO FIRE
C              - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C              - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C     TTRACK(JM)
C     XT(IM,2)
C     XXINT(JM)
C     YT(IM,2)
C     ZT(IM,2)
C.....LOCAL TO THIS SUBROUTINE.....
C     IK
C.....START PROGRAM.....

C     WRITE(*,*) 'FIRE2'

C.....TRACK TARGET AND ASSESS INTERCEPT.....

      I=ITAR(J)
      RRAD(J)=RRAD(J)+DT

C.....TOO EARLY FOR INTERCEPT.....

C.....CHECK IF PEN. IS TERRAIN MASKED TOO LONG.....

      IF (RRAD(J) .LT. XXINT(J)+TTRACK(J)) THEN
        IF (MASKTIM(J) .EQ. 0.0) THEN
          CALL TERRA(IK)
          IF (IK .EQ. 1) THEN
            MASKTIM(J)=TIME
            IF (DEBUG .EQ. 1) THEN
              WRITE (16,1007) TIME,I,J,XT(I,2),YT(I,2),ZT(I,2)
1007            FORMAT(F5.0,' SEC PEN. ',I2,' MASKED FROM SITE ',I2,
     $          ' BEFORE SAM INTERCEPTS ',3F6.2)
            ENDIF
            WRITE(17,2007) 73,1
2007          FORMAT(I2,I2)
            WRITE(17,3007) TIME,I,J,0,0,0.0
3007          FORMAT(F5.0,I2,I2,I3,I3,F9.4)
          ENDIF
        ELSEIF(MASKTIM(J) .NE. 0.0) THEN
          CALL TERRA(IK)
```

```
          IF (IK .EQ. 1) THEN
            IF(TIME-MASKTIM(J) .EQ. MWAITIM(J)) THEN
              MASKTIM(J)=0.0
              STATS(J)=0
              IF (DEBUG .GE. 1) THEN
                WRITE(16,1005) TIME,J,I,XT(I,2),YT(I,2),ZT(I,2)
1005              FORMAT(F5.0,' SEC SITE ',I2,' LOSES TRACK OF PEN. '
     $           ,I2,' :TERRAIN MASKED TOO LONG ',3F6.2)
              ENDIF
              WRITE(17,2005) 71,1
2005           FORMAT(I2,I2)
              WRITE(17,3005) TIME,I,J,0,0,0.0
3005           FORMAT(F5.0,I2,I2,I3,I3,F9.4)
            ENDIF
          ELSEIF (IK .NE. 1) THEN
            MASKTIM(J)=0.0
            IF (DEBUG .EQ. 1) THEN
              WRITE (16,1009) TIME,I,J,XT(I,2),YT(I,2),ZT(I,2)
1009           FORMAT(F5.0,' SEC PEN. ',I2,' UMMASKED FROM SITE ',I2,
     $         ' BEFORE SAM INTERCEPTS ',3F6.2)
            ENDIF
            WRITE(17,2009) 75,1
2009         FORMAT(I2,I2)
            WRITE(17,3009) TIME,I,J,0,0,0.0
3009         FORMAT(F5.0,I2,I2,I3,I3,F9.4)
          ENDIF
        ENDIF

C.....CHECK IF PEN. IS OUT OF ENVELOPE FOR TOO LONG.....


        IF (OUTIM(J) .EQ. 0.0) THEN
          IF (CD1(I,J) .GT. 20000.0) THEN
            OUTIM(J)=TIME
            IF (DEBUG .EQ. 1) THEN
              WRITE (16,1008) TIME,I,J,XT(I,2),YT(I,2),ZT(I,2)
1008           FORMAT(F5.0,' SEC PEN. ',I2,' OUT OF ENVELOPE OF SITE'
     $         ,' ',I2,' BEFORE SAM INTERCEPTS ',3F6.2)
            ENDIF
            WRITE(17,2008) 74,1
2008         FORMAT(I2,I2)
            WRITE(17,3008) TIME,I,J,0,0,0.0
3008         FORMAT(F5.0,I2,I2,I3,I3,F9.4)
          ENDIF
        ELSEIF(OUTIM(J) .NE. 0.0) THEN
          IF (CD1(I,J) .GT. 20000.0) THEN
            IF (TIME-OUTIM(J) .EQ. OWAITIM(J)) THEN
              OUTIM(J)=0.0
              STATS(J)=0
              IF (DEBUG .GE. 1) THEN
```

```
                WRITE(16,1006) TIME,J,I,XT(I,2),YT(I,2),ZT(I,2)
1006             FORMAT(F5.0,' SEC SITE ',I2,' LOSES TRACK OF PEN. '
   $            ,I2,':OUT OF ENVELOPE TOO LONG ',3F6.2)
               ENDIF
               WRITE(17,2006) 72,1
2006            FORMAT(I2,I2)
               WRITE(17,3006) TIME,I,J,0,0,0.0
3006            FORMAT(F5.0,I2,I2,I3,I3,F9.4)
             ENDIF
           ELSEIF (
     $(CD4(I,J) .GE. 20000.0 .AND.
     $(TIME. GE. CD1(I,J) .AND. TIME .LE. CD2(I,J)))
     $.OR.
     $(CD4(I,J) .LT. 20000.0 .AND.
     $(TIME .GE. CD1(I,J) .AND. TIME .LE. CD2(I,J)) .OR.
     $(TIME .GE. CD3(I,J) .AND. TIME .LE. CD4(I,J)))) THEN
             OUTIM(J)=0.0
             IF (DEBUG .GE. 1) THEN
               WRITE(16,1010) TIME,J,I,XT(I,2),YT(I,2),ZT(I,2)
1010            FORMAT(F5.0,' SEC PEN. ',I2,' BACK INSIDE ENVELOPE',
   $           ' OF SITE ',I2,' BEFORE SAM INTERCEPTS ',3F6.2)
             ENDIF
             WRITE(17,2010) 76,1
2010          FORMAT(I2,I2)
             WRITE(17,3010) TIME,I,J,0,0,0.0
3010          FORMAT(F5.0,I2,I2,I3,I3,F9.4)
           ENDIF
         ENDIF
         IF (STATS(J) .EQ. 0) THEN
           MASKTIM(J)=0.0
           OUTIM(J)=0.0
         ENDIF
         GO TO 999
       ENDIF

       MASKTIM(J)=0.0
       OUTIM(J)=0.0

C.....3 MEANS POST-SHOT ACTIVITY.....


       STATS(J)=3
       RRAD(J)=GDT(J)

       IF (TIME .GT. CD2(I,J)  .AND.  CD3(I,J) .GT. 20000.  .OR.
      $TIME .GT. CD4(I,J)) GO TO 130
       IF (CD3(I,J) .LT. 20000.  .AND.  TIME .GT. CD2(I,J)  .AND.
      $TIME .LT. CD3(I,J)) GO TO 140

C.....12/18/87 NEW TEST CODE BFS.....
```

```
      IF (CD1(I,J) .GT. 20000.  .AND.  TIME .LT. CD1(I,J))
     $GO TO 150
      CALL TERRA(IK)
      IF (IK .EQ. 1) GO TO 160

       CALL ASSESS
       GO TO 999

C.....

  130 IF (DEBUG .GE. 1) THEN
         WRITE(16,1001) TIME,J,I,XT(I,2),YT(I,2),ZT(I,2)
         WRITE(16,1011) CD1(I,J),CD2(I,J),CD3(I,J),CD4(I,J)
 1001 FORMAT(F5.0,' SEC SITE ',I2,' MISSES PEN. ',I2, ' > MAX RANGE ',
     $   3F8.2)
 1011 FORMAT(' CD1-CD4: ',4F9.2)
      ENDIF
      WRITE(17,2001) 27,1
 2001 FORMAT(I2,I2)
      WRITE(17,3001) TIME,I,J,0,0,0.0
 3001 FORMAT(F5.0,I2,I2,I3,I3,F9.4)

      GO TO 170

C.....

  140 IF (DEBUG .GE. 1) THEN
         WRITE(16,1002) TIME,J,I,XT(I,2),YT(I,2),ZT(I,2),CD1(I,J),
     $   CD2(I,J),CD3(I,J),CD4(I,J)
 1002 FORMAT(F5.0,' SEC SITE ',I2,' MISSES PEN. ',I2, ' < MIN. RANGE '
     $   ,3F8.2,' CD1-CD4: ',4F9.2)
      ENDIF
      WRITE(17,2002) 26,1
 2002 FORMAT(I2,I2)
      WRITE(17,3002) TIME,I,J,0,0,0.0
 3002 FORMAT(F5.0,I2,I2,I3,I3,F9.4)

      GO TO 170

C.....

  150 IF (DEBUG .GE. 1) THEN
         WRITE(16,1003) TIME,J,I,XT(I,2),YT(I,2),ZT(I,2),CD1(I,J),
     $   CD2(I,J),CD3(I,J),CD4(I,J)
 1003 FORMAT(F5.0,' SEC SITE ',I2,' MISSES PEN. ',I2,' OUT OF ENVELOPE '
     $   ,3F8.2,' CD1-CD4: ',4F9.2)
      ENDIF
      WRITE(17,2003) 69,1
 2003 FORMAT(I2,I2)
      WRITE(17,3003) TIME,I,J,0,0,0.0
```

```
3003 FORMAT(F5.0,I2,I2,I3,I3,F9.4)


     GO TO 170

C.....

 160 IF (DEBUG .GE. 1) THEN
        WRITE(16,1004) TIME,J,I,XT(I,2),YT(I,2),ZT(I,2)
1004    FORMAT(F5.0,' SEC SITE ',I2,' MISSES PEN. ',I2,
    $   ' :TERRAIN MASKED ',3F8.2)
     ENDIF
     WRITE(17,2004) 70,1
2004 FORMAT(I2,I2)
     WRITE(17,3004) TIME,I,J,0,0,0.0
3004 FORMAT(F5.0,I2,I2,I3,I3,F9.4)


     GO TO 170

C.....


 170 CONTINUE

 999 RETURN
     END
---------------------------------
     SUBROUTINE FIRE3

C.....LATEST CHANGE MADE  6/28/91 BFS.....
C     COPY OF SHOT3 FROM NON-IADS MODE
C
C     CALLED BY - FIRE
C     CALLS     - TERRA,JAMMER,NTRPCT,FUTLOK
C.....

     PARAMETER (IM=30,JM=25,KM=20)

     IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

     REAL TF
     INTEGER FUTSTATS,IK,K,MFLG

C.....DECLARE VARIABLES IN COMMON BLOCK.....

     REAL CD1,CD2,CD3,CD4,DT,PSA,PSAMIN,RRAD,TIME,TTRACK
     INTEGER DEBUG,I,ISECT,ISHOT,ITAR,J,MSITE,NAC,NOSHOT,NSHOT,STATS
```

```
C.....COMMON BLOCK.....

      COMMON/CRUISE/IOP(10),TIMEL,RRAD(JM),IT(JM)
      COMMON/DEBUG/DEBUG
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/RCTIME/TRACK(9),TTRACK(JM),NUMSHT(JM),NOSHOT(JM),NSHOT(JM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....IN COMMON BLOCK.....
C     CD1(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM ENTERS OUTER
C                - LAUNCH ENVLOPE OF SAM SITE JM
C     CD2(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS IS
C                - THE ENVELOPE EXIT TIME. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE EXIT OF THE ENVELOPE INTO THE SITES
C                - MIN RANGE ZONE.
C     CD3(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C                - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE ENTRANCE INTO THE ENVELOPE FROM THE
C                - SITES MIN RANGE ZONE.
C     CD4(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C                - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE EXIT OF THE PENETRATOR FROM THE ENVELOPE
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     DT - TACOPS TIME INCREMENT (1 SECOND)
C     I
C     ISECT(IM,JM) - HOLDS A FLAG(1/0) FOR EACH PENETRATOR IM WHICH
C                  - INDICATES WHETHER EACH SAM SITE JM CAN(=1) OR
C                  - CANNOT(=0) FIRE AT THIS PENETRATOR
C     ISHOT(IM,JM)
C     ITAR(JM) - HOLDS PEN. # THAT SAM SITE JM IS LOCKED ONTO
C     J - SAM SITE LOOP COUNTER
C     MSITE(JM) - HOLDS THE SAM SITE TYPE (1-9) FOR SITE JM
```

```
C     NAC - MAX # OF PENETRATORS
C     NOSHOT(JM) - MAX # OF SALVOS AVAILABLE TO SAM SITE (JM)
C     NSHOT(JM) - RUNNING TOTAL # OF SALVOS SAM SITE JM HAS FIRED
C     PSA(IM) - SURVIVAL PROBABILITY OF PENETRATOR IM
C     PSAMIN - MINIMUM SURVIVABILITY OF PENETRATOR;IF PSA(IM) IS BELOW
C           - THIS, THE PENETRATOR IS REMOVED FROM THE SIMULATION
C     RRAD(JM) - HOLDS TIMING DATA (IN SECONDS) FOR SAM SITE JM; USED
C              - IN DETERMINING WHEN TO FIRE,RE-FIRE,TRACK,ETC...
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C               - IF =0; SITE IS NOT RADIATING
C               - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C               - AND GETTING READY TO FIRE
C               - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C               - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C     TTRACK(JM)
C.....LOCAL TO THIS SUBROUTINE.....
C     FUTSTATS - FUTURE STATUS OF SAM SITE. =1 IF SITE WILL CONTINUE TO
C              - TO RADIATE. =0 IF NO OTHER TARGETS ARE PRESENTLY WITHIN
C              - RANGE TO LOCKON TO.
C     IK - FLAG(1/0) USED BY SUBROUTINE JAMMER TO INDICATE IF(=1)
C        - THE PENETRATOR IS MASKED TO A SAM SITE BY JAMMING; BY
C        - SUBROUTINE TERRA TO INDICATE IF(=1) THE PENETRATOR IS MASKED
C        - BY THE TERRAIN FROM A SAM SITE; BY SUBROUTINE CHECKS TO
C        - INDICATE(=1) THAT THE PENETRATOR HAS BEEN DESTROYED; BY
C        - SUBROUTINE FUTLOK TO SEE IF(=1) THE ENGAGEMENT ANGLE OF THE
C        - PENETRATOR AND THE SAM IS WITHIN THE VALUE TO ALLOW FOR A
C        - LAUNCH AGAINST THIS PENETRATOR
C     K -
C     MFLG - FLAG TO INDICATE IF SITE WILL CONTINUE TO RADIATE OR SHUT
C          - DOWN
C     TF -
C.....START PROGRAM.....

C     WRITE(*,*) 'FIRE3'

C.....SHOT'S OVER.  NOW WHAT?.....

      RRAD(J)=RRAD(J)-DT

      IF (RRAD(J) .GT. DT) GO TO 99


C.....BEGIN NEW CODE 10/27/87 BFS .....

      MFLG=0
      FUTSTATS=1

      DO 70 I=1,NAC
      IF (ISECT(I,J) .EQ. 0  .OR. PSA(I) .LT. PSAMIN) GO TO 70
```

```
      CALL TERRA(IK)
      IF (IK .NE. 0) GO TO 70
      CALL JAMMER(IK)
      IF (IK .NE. 0) GO TO 70

C.....CHECK FOR PENETRATOR WITHIN SAM CONE.....
C.....NO INTERSECTION WITH CONE.....

      IF (CD1(I,J) .GT. 20000.) GO TO 70
      IF (TIME+1 .LT. CD1(I,J)-TTRACK(J)  .OR. TIME+1 .GT. CD4(I,J)-
     $TTRACK(J)) GO TO 70
      IF (TIME+1+TTRACK(J) .LT. CD2(I,J)) GO TO 80
      IF (CD3(I,J) .GT. 20000.) GO TO 70
      IF (TIME+1+TTRACK(J) .GT. CD2(I,J)  .AND.  TIME+1+TTRACK(J) .LT.
     $CD3(I,J)) GO TO 70

   80 CALL FUTLOK(IK)
      IF (IK .NE. 0) GO TO 70

   90 MFLG=1

   70 CONTINUE

C.....0 MEANS SITE J STOPS RADIATING.....

      IF (MFLG .EQ. 0) THEN
        FUTSTATS=0
      ENDIF

C..... END NEW CODE 10/27/87 BFS.....

      I=ITAR(J)
      STATS(J)=0

      IF (NSHOT(J) .EQ. NOSHOT(J)) NSHOT(J)=NSHOT(J)+1

C.....NO MORE SHOTS FOR THE SAM SITE IF: THE SITE Ps IS LESS THAN
C.....THAT GIVEN OR IT HAS SHOT ALL ITS SALVOS.....

      IF (PSA(I) .LT. PSAMIN  .OR.  NSHOT(J) .GT. NOSHOT(J)) GO TO 199

      CALL FUTLOK(IK)
      IF (IK .NE. 0) GO TO 199

C.....TAKE THE SHOT?.....

C.....FOOL TARGET TRACK ROUTINE.....

  180 RRAD(J)=TTRACK(J)-DT
```

```
      CALL NTRCPT(TF)

C.....CHECK TO SEE IF TARGET IS IN THE ENVELOPE  (NEW 12/18/87)....

      IF (CD1(I,J) .GT. 20000.) GO TO 199

C.....END NEW CODE 12/18/87.....
C.....TAKE THE SHOT.....

      IF (TIME+TF .LT. CD2(I,J)) THEN
      GO TO 190
      ENDIF

C.....TARGETS MAX RANGE.....

      IF (CD3(I,J) .GT. 20000.  .OR.  TIME+TF .GT. CD4(I,J)) GO TO 199

C.....BETTER TO GET NEW TARGET.....

      IF (CD3(I,J)-TIME .GT. TTRACK(J)) GO TO 199

C.....MAYBE WAIT FOR REAR SHOT.....

      IF (TIME .LT. CD3(I,J)) RRAD(J)=TIME+TTRACK(J)-CD3(I,J)

  190 STATS(J)=1


      DO 200 K=1,NAC
      ISHOT(K,J)=0
  200 CONTINUE

  199 CONTINUE


      IF (STATS(J) .EQ. 0 ) ITAR(J)=0


      IF (STATS(J) .EQ. 0  .AND. FUTSTATS .EQ. 0) THEN
        IF (DEBUG .GE. 1) THEN
          WRITE(16,1000) TIME,J
 1000       FORMAT(F5.0,' SEC SITE ',I2,' STOPS RADIATING')
        ENDIF
        WRITE(17,2000) 52,1
 2000     FORMAT(I2,I2)
        WRITE(17,3000) TIME,0,J,0,0,0.0
 3000     FORMAT(F5.0,I2,I2,I3,I3,F9.4)
      ENDIF
```

```
  99 RETURN
     END
--------------------------------
     SUBROUTINE FIRE4

C.....LATEST CHANGE MADE  6/28/91 BFS.....
C     COPY OF SHOT4 FROM NON-IADS MODE
C
C     CALLED BY - FIRE1
C     CALLS     - CHECKS,SUSPNS
C.....

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER IK

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      REAL CD1,CD2,CD3,CD4,RRAD,TIME,TTRACK,XT,YT,ZT
      INTEGER DEBUG,I,IX,J,STATS

C.....COMMON BLOCK.....

      COMMON/CRUISE/IOP(10),TIMEL,RRAD(JM),IT(JM)
      COMMON/DEBUG/DEBUG
      COMMON/RCTIME/TRACK(9),TTRACK(JM),NUMSHT(JM),NOSHOT(JM),NSHOT(JM)
      COMMON/SHOTA/XXINT(JM),IX(JM),MASKTIM(JM),MWAITIM(JM),OUTIM(JM),
     $OWAITIM(JM)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....IN COMMON BLOCK.....
C     CD1(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM ENTERS OUTER
C                - LAUNCH ENVLOPE OF SAM SITE JM
C     CD2(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C                - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C                - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS IS
C                - THE ENVELOPE EXIT TIME. IF 4 INTERSECTIONS OCCUR,
C                - THIS IS THE EXIT OF THE ENVELOPE INTO THE SITES
```

```
C             - MIN RANGE ZONE.
C     CD3(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C             - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C             - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C             - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C             - THIS IS THE ENTRANCE INTO THE ENVELOPE FROM THE
C             - SITES MIN RANGE ZONE.
C     CD4(IM,JM) - TIME (IN SECONDS) WHEN PENETRATOR IM TRANSITIONS
C             - LAUNCH ENVLOPE OF SAM SITE JM. IF THE PENETRATORS
C             - FLIGHTPATH INTERSECTS THE ENVELOPE 2 TIMES, THIS
C             - TIME HAS NO MEANING. IF 4 INTERSECTIONS OCCUR,
C             - THIS IS THE EXIT OF THE PENETRATOR FROM THE ENVELOPE
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     I - PEN. LOOP COUNTER
C     IX(JM) - FLAG (1/0) USED TO CONTROL PRINTING OUTPUT DATA
C     J - SAM SITE LOOP COUNTER
C     RRAD(JM) - HOLDS TIMING DATA (IN SECONDS) FOR SAM SITE JM; USED
C           - IN DETERMINING WHEN TO FIRE,RE-FIRE,TRACK,ETC...
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C             - IF =0; SITE IS NOT RADIATING
C             - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C             - AND GETTING READY TO FIRE
C             - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C             - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C     TTRACK(JM)
C     XT(IM,2)
C     YT(IM,2)
C     ZT(IM,2)
C.....LOCAL TO THIS SUBROUTINE.....
C     IK - FLAG(1/0) USED BY SUBROUTINE JAMMER TO INDICATE IF(=1)
C       - THE PENETRATOR IS MASKED TO A SAM SITE BY JAMMING; BY
C       - SUBROUTINE TERRA TO INDICATE IF(=1) THE PENETRATOR IS MASKED
C       - BY THE TERRAIN FROM A SAM SITE; BY SUBROUTINE CHECKS TO
C       - INDICATE(=1) THAT THE PENETRATOR HAS BEEN DESTROYED; BY
C       - SUBROUTINE FUTLOK TO SEE IF(=1) THE ENGAGEMENT ANGLE OF THE
C       - PENETRATOR AND THE SAM IS WITHIN THE VALUE TO ALLOW FOR A
C       - LAUNCH AGAINST THIS PENETRATOR
C.....START PROGRAM.....

C     WRITE(*,*) 'FIRE4'

C.....DOES INTRUDER EVADE SAM SYSTEM?.....

      CALL CHECKS(IK)
      IF (IK .NE. 0) GO TO 99

C.....WITHIN CONE.....

      IF (TIME .LT. CD2(I,J)) GO TO 99
```

```
C.....MUST BE LONG.....

      IF (CD3(I,J) .GT. 20000.) GO TO 160

C.....NOT LONG.....

      IF (TIME .LT. CD4(I,J)) GO TO 170

C.....PRINT MESSAGE ONCE.....

  160 IF (IX(J) .NE. 0) GO TO 180
      IX(J)=1
      IF (DEBUG .GE. 1) THEN
        WRITE(16,1005) TIME,J,I,XT(I,2),YT(I,2),ZT(I,2),CD1(I,J),
     $   CD2(I,J),CD3(I,J),CD4(I,J)
 1005    FORMAT(F5.0,' SEC SITE ',I2,' WILL LOSE PEN. ',I2,
     $   ' IN 3 SECS. - LONG ',3F7.1,'CD1-CD4: ',4F8.1)
      ENDIF
C.....WRITE EVENT AND DATA TO DOC FILE.....

      WRITE(17,2001) 28,1
 2001 FORMAT(I2,I2)
      WRITE(17,3001) TIME,I,J,0,0,0.0
 3001 FORMAT(F5.0,I2,I2,I3,I3,F9.4)

C.....LONG GONE.....

  180 CALL SUSPNS
      GO TO 99

C.....ISN'T SHORT.....

  170 IF (TIME .GT. CD3(I,J)) GO TO 99

C.....PERFORM DECISION ONLY ONCE.....

      IF (IX(J) .NE. 0) GO TO 99
      IX(J)=1

C.....LOCK ON NEW TARGET BEST.....

      IF (CD3(I,J)-TIME .GT. TTRACK(J)) THEN
        STATS(J)=0
        IF (DEBUG .GE. 1) THEN
          WRITE(16,1000) TIME,J
 1000       FORMAT(F5.0,' SEC SITE ',I2,' STOPS RADIATING')
        ENDIF
        WRITE(17,2000) 52,1
 2000    FORMAT(I2,I2)
        WRITE(17,3000) TIME,0,J,0,0,0.0
```

```
 3000    FORMAT(F5.0,I2,I2,I3,I3,F9.4)
      ENDIF
C.....SAM FIRES IF STATS IS UNCHANGED.....

      RRAD(J)=TIME+TTRACK(J)-CD3(I,J)

      IF (DEBUG .GE. 1) THEN
        WRITE(16,1006) TIME,J,I,XT(I,2),YT(I,2),ZT(I,2),CD1(I,J),
     $   CD2(I,J),CD3(I,J),CD4(I,J)
 1006    FORMAT(F5.0,' SEC SITE ',I2,' HAS PEN. ',I2,
     $   ' IN MIN RANGE ',3F7.1,'CD1-CD4: ',4F8.1)
      ENDIF
C.....WRITE EVENT AND DATA TO DOC FILE.....

      WRITE(17,2002) 29,1
 2002 FORMAT(I2,I2)
      WRITE(17,3002) TIME,I,J,0,0,0.0
 3002 FORMAT(F5.0,I2,I2,I3,I3,F9.4)

   99 RETURN
      END
---------------------------------
      SUBROUTINE GETDET(ORIGUNIT,MESID,MSG2)

C.....LATEST CHANGE MADE 07/19/91 BFS.....
C     FOR MESSAGES GOING TO ADWOC CENTER FROM AN EW SITE. THIS SUB
C     LOCATES AND RECEIVES THE DETECTION DATA ASSOCIATED WITH THE
C     MESSAGE AND UPDATES THE ADWOCS KNOWN TARGET DETECTION STATUS.
C
C     CALLED BY - RADWOC
C     CALLS     - NONE
C.....

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

      INTEGER ORIGUNIT,MESID
      REAL MSG2

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG,NAC
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)

C.....COMMON BLOCK.....
```

```
      INCLUDE 'QIADS  DEF'
      INCLUDE 'HOLDM1 DEF'
      INCLUDE 'KNOWN1 DEF'

      COMMON/DEBUG/DEBUG

C.....DECLARE LOCAL VARIABLES.....

      INTEGER JJ,L,LL,II,DATAPOS,DATA(IM),ENDPOS

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                          .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     NAC - MAX # OF PENETRATORS
C.....LOCAL TO THIS SUBROUTINE.....
C     DATAPOS - POSITION OF ASSOCIATED DATA
C     DATA - HOLDS DETECTION DATA ASSOCIATED WITH RECEIVED MESSAGE
C     ENDPOS - ENDING POSITION OF EW SITE MSGID # ARRAY
C     II - PASSED HOLDING MESSAGE ARRAY POINTER
C     JJ - LOOP COUNTER
C     L - LOOP COUNTER
C     LL - LOOP COUNTER
C     MSG2 - HOLDS RECEIVED MESSAGE TIME
C     MESID - INCOMING MESSAGE ID #
C     ORIGTYPE - INTEGER HOLDING THE ORIGIN SOURCE MESSAGE IADS TYPE #
C     ORIGUNIT - INTEGER HOLDING THE ORIGIN SOURCE MESSAGE IADS UNIT #
C     POS - POSITION OF NEXT OPEN SLOT IN MESSAGE ID HOLDING ARRAY
C     PREVTYPE - INTEGER HOLDING THE PREVIOUS SOURCE IADS TYPE #
C     UNIT - HOLDS UNIT #
C.....START PROGRAM.....

C     WRITE(*,*) 'GETDET'

C.....LOCATE POSITION OF EW DETECTION DATA CORRELATING ON MESID.....

      DO 17 JJ=1,EWMESPOS(ORIGUNIT)-1
      IF(NINT(EWDETMES(ORIGUNIT,9,JJ)) .EQ. MESID) THEN
        DATAPOS=JJ
        GOTO 18
      ENDIF
   17 CONTINUE
   18 CONTINUE

      IF(DATAPOS .EQ. 0) THEN
        WRITE(*,*)  'POSITION OF EW DETECTION DATA NOT FOUND',
```

```
     & ',STOPPING PROGRAM IN SUB GETDET'
        WRITE(16,*) 'POSITION OF EW DETECTION DATA NOT FOUND',
     & ',STOPPING PROGRAM IN SUB GETDET'
       ENDIF

       IF (DEBUG .GE. 2) THEN
         WRITE(16,*) 'DATA IS AT POINTER POSITION',DATAPOS
         WRITE(16,*) 'HOLDING MESSAGE ID # IS NOW =',MESID
       ENDIF

C... TRANSFER DATA IN .....

       DO 20 L=1,NAC
       DATA(L)=EWDETDAT(ORIGUNIT,L,DATAPOS)
  20   CONTINUE

       IF (DEBUG .GE. 2) THEN
         WRITE(16,*) 'MESSAGE DATA FROM EW SITE #',
     &    NINT(EWDETMES(ORIGUNIT,5,DATAPOS)), ' MESSAGE #',DATAPOS
         WRITE(16,*)'EWDETMES(ORIGUNIT,L);L=1,?',
     &   (EWDETMES(ORIGUNIT,L,DATAPOS),L=1,MAXMSGSIZE)
         WRITE(16,*) 'EWDETDAT(ORIGUNIT,L,DATAPOS);L=1,NAC',
     &       (EWDETDAT(ORIGUNIT,L,DATAPOS),L=1,NAC)
       ENDIF

C... SHIFT PENDING MESSAGES UP ONE POSITION, CLEAR DATA AT LAST
C ... POINTER POSITION AND SET POINTER OF ONE POSITION LESS.....

       ENDPOS=EWMESPOS(ORIGUNIT)

       DO 30 L=DATAPOS,ENDPOS
       DO 40 II=1,MAXMSGSIZE
       EWDETMES(ORIGUNIT,II,L-1)=EWDETMES(ORIGUNIT,II,L)
  40   CONTINUE
       DO 50 II=1,NAC
       EWDETDAT(ORIGUNIT,II,L-1)=EWDETDAT(ORIGUNIT,II,L)
  50   CONTINUE
  30   CONTINUE
       EWMESPOS(ORIGUNIT)=ENDPOS-1
       IF(DEBUG .GE. 3) THEN
         WRITE(16,*) '   INCOMING EW MESSAGE POINTER DECREMENTED',
     &  ' TO =', EWMESPOS(ORIGUNIT)
       ENDIF

C ... PROCESS THE NEW DETECTION DATA WITH THE EXISTING INFORMATION ...
C ... UPDATE THE KNOWN INFORMATION FOR THE EW SITE AND FOR THE COMBINED
C ... DETECTION KNOWLEDGE OF ALL TARGETS.....

       DO 60 LL=1,NAC
       DETTGT(ORIGUNIT,LL)=DATA(LL)+DETTGT(ORIGUNIT,LL)
```

```
  60 CONTINUE

C ... CLEAR OLD DATA AND RECOMPUTE NEW DETECTION DATA .....

     DO 65 LL=1,NAC
     DETALL(LL)=0
  65 CONTINUE

     DO 70 JJ=1,NEWSITES
     DO 80 LL=1,NAC
     DETALL(LL)=DETTGT(JJ,LL)+DETALL(LL)
  80 CONTINUE
  70 CONTINUE

     IF(DEBUG .GE. 3) THEN
       WRITE(16,1003) MSG2,ORIGUNIT
1003   FORMAT(F5.0,' SEC RECEIVED MESSAGE UPDATE FROM EW SITE ',I2)
       WRITE(16,1004) (DATA(LL),LL=1,NAC)
1004   FORMAT(10X,'FOR THIS EW SITE, UPDATE MESSAGE IS      ',50I3)
       WRITE(16,1005) (DETTGT(ORIGUNIT,LL),LL=1,NAC)
1005   FORMAT(10X,'FOR THIS EW SITE, NEW PEN. DET. STATUS IS',50I3)
     ENDIF
     IF(DEBUG .GE. 3) THEN
       WRITE(16,*) '         NEW DETECTION STATUS FOR ALL SITES:'
       DO 100 JJ=1,NEWSITES
       WRITE(16,1007) JJ,(DETTGT(JJ,LL),LL=1,NAC)
1007   FORMAT(10X,'EW SITE #',I3,5X,50I3)
 100   CONTINUE
     ENDIF
     IF(DEBUG .GE. 3) THEN
       WRITE(16,*)' AT ADWOC CENTER, COMBINED DET. STATUS IS'
       WRITE(16,1006) (DETALL(LL),LL=1,NAC)
1006   FORMAT(27X,50I3)
     ENDIF

     RETURN
     END
---------------------------------
     SUBROUTINE HOLDMES(MSG)

C.....LATEST CHANGE MADE 07/09/91 BFS.....
C     RECEIVES A MESSAGE AND PUTS IT ON THE WAITING MESSAGE ARRAY
C
C     CALLED BY - EWMES,REW,REWSTA,ADWOC,INITEW,RBAT,RSAM
C     CALLS     - NONE
C.....

     PARAMETER (IM=30,JM=25,KM=20)
```

```
C.....DECLARE LOCAL VARIABLES.....

      INTEGER II,POS

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG

C.....COMMON BLOCK.....
      INCLUDE 'HOLDM1 DEF'

      COMMON/DEBUG/DEBUG

C.....DECLARE PASSED VARIABLES .....

      REAL MSG(MAXMSGSIZE)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                          .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C.....LOCAL TO THIS SUBROUTINE.....
C     II - LOOP COUNTER
C     POS - INPUT INDEX OF MESSAGE IN HOLDING ARRAY
C.....START PROGRAM.....

C     WRITE(*,*) 'HOLDMES'

      POS=NEXTHOLDPOS

      IF(POS .GT. 200) THEN
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(*,*) 'MESHOLD ARRAY IS FULL AND THERE ARE MORE',
     &   ' MESSAGES'
        WRITE(16,*) 'MESHOLD ARRAY IS FULL AND THERE ARE MORE',
     &   ' MESSAGES'
        WRITE(*,*) 'PRESENT DIMENSION OF MESHOLD IS 200'
        WRITE(*,*) 'INCREASE DIMENSION IN HOLDM1 DEF AND',
     &   ' RECOMPILE PROGRAM MODULES'
        WRITE(16,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
      STOP
      ENDIF

      IF(DEBUG .GE. 3) THEN
        WRITE(16,*) 'PRESENT HOLDING MESSAGE POSITION = ',
```

```
     &   NEXTHOLDPOS
      ENDIF

      DO 10 II=1,MAXMSGSIZE
      MESHOLD(POS,II)=MSG(II)
   10 CONTINUE

      IF(DEBUG .GE. 3) THEN
        WRITE(16,*) 'NEW MESSAGE PUT IN HOLDING ARRAY IS:',
     &    (MESHOLD(POS,II),II=1,MAXMSGSIZE)
      ENDIF

      NEXTHOLDPOS=NEXTHOLDPOS+1

      IF(DEBUG .GE. 2) THEN
        WRITE(16,*) 'NEXT HOLDING MESSAGE POSITION = ',
     &   NEXTHOLDPOS
      ENDIF

      RETURN
      END
---------------------------------
      SUBROUTINE IADS

C.....LATEST CHANGE MADE 7/03/91 BFS.....
C     CONTROLS DETECTION OF AND FIRING ON PENETRATORS AND CALLING OF
C     IADS SUBROUTINES
C
C     CALLED BY - TACOPS
C     CALLS     - INITEW,EWDET,FIRE,MESRDY
C.....

      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE VARIABLES IN COMMON BLOCK.....


      REAL PS,PSSMIN,TIME
      INTEGER DEBUG,ITAR,J,NSITE,NOSHOT,NSHOT,STATS
      INTEGER IOP
C.....COMMON BLOCK.....

      INCLUDE 'EW1 DEF'
      COMMON/CRUISE/IOP(10),TIMEL,RRAD(JM),IT(JM)

      COMMON/DEBUG/DEBUG
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
```

```
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/RCTIME/TRACK(9),TTRACK(JM),NUMSHT(JM),NOSHOT(JM),NSHOT(JM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     ITAR(JM) - HOLDS PEN. # THAT SAM SITE JM IS LOCKED ONTO
C     J - SAM SITE LOOP COUNTER
C     NSITE - MAX # OF SAM SITES
C     NOSHOT(JM) - MAX # OF SALVOS AVAILABLE TO SAM SITE (JM)
C     NSHOT(JM) - RUNNING TOTAL # OF SALVOS SAM SITE JM HAS FIRED
C     PS(JM) - SURVIVAL PROBABILITY OF SAM SITE JM
C     PSSMIN - MIN SURVIVAL PROBABILITY THAT THE SAM SITE MUST MAINTAIN
C           - IN ORDER TO STAY IN THE SIMULATION
C     START - FLAG(1/0) =1 IMPLIES THIS THE STARTING (FIRST) LOOP
C           - OF THE SIMULATION. DO INITIALIZING THINGS
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C               - IF =0; SITE IS NOT RADIATING
C               - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C               - AND GETTING READY TO FIRE
C               - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C               - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C.....START PROGRAM.....

C     WRITE(*,*) 'IADS'

C ... AT START OF SIMULATION, SET THE INITIAL STATUS OF EACH EW SITE..

      IF(START .EQ. 1) THEN
        START=0
        CALL INITEW
      ENDIF

C     IF(NINT(TIME) .EQ. 260) CALL EWOFF
C     IF(NINT(TIME) .EQ. 101) CALL EWOFF
C     IF(NINT(TIME) .EQ. 400) CALL SAMOFF
C     IF(NINT(TIME) .EQ. 20) CALL SAMON
      CALL EWDET
      CALL MESRDY
```

```
C     CALL ALLOCATE TARGETS
      CALL FIRE

      IF(TIME .GT. 1999.) THEN
        WRITE(*,*) 'STOPPING SIM (IN SUB IADS) AT TIME=2000'
        STOP
      ENDIF

      RETURN
      END
-----------------------------------
      SUBROUTINE IADSPS(TYPE,UNIT1,UNIT2,RND1)

C.....LATEST CHANGE MADE 07/30/91 BFS.....
C     WHEN AN ELEMENT OF THE IADS STRUCTURE IS DAMAGED, THIS SUBROUTINE
C     SCANS THROUGH THE HOLDING MESSAGE ARRAY, LOCATES ANY MESSAGES
C       ASSOCIATED WITH THE NEWLY DAMAGED ELEMENT AND UPDATED THE
SENDING
C     TIME OF THE MESSAGE BASED UPON THE NEW DAMAGE LEVEL.
C
C     CALLED BY - TICK2
C     CALLS     - DELMES,CDELAY,DLYMES,EWDET2,EWMES
C.....

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

      INTEGER TYPE,UNIT1,UNIT2
      REAL RND1
C.....DECLARE LOCAL VARIABLES.....

      INTEGER IADSTYPE,IADSUNIT,II,JJ,LL,L,UPDATE(IM)
      REAL DELAY

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG,I,J,NAC
      REAL PS,PSSMIN,PST,TIME,XS,YS

C.....COMMON BLOCK.....

      INCLUDE 'EWDATA  DEF'
      INCLUDE 'RPTDATA DEF'
      INCLUDE 'ADWDATA DEF'
      INCLUDE 'BATDATA DEF'
      INCLUDE 'SAMDATA DEF'
      INCLUDE 'HOLDM1  DEF'
```

```
      INCLUDE 'QIADS   DEF'
      INCLUDE 'CON1    DEF'
      INCLUDE 'EW1     DEF'

      COMMON/DEBUG/DEBUG
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     I - PENETRATOR #
C     J - SAM SITE #
C     NAC - MAX # OF PENETRATORS
C     PS(JM) - SURVIVAL PROBABILITY OF SITE JM
C              - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C              - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     PSSMIN - MIN. SURVIVAL PROBABILITY FOR SAM SITE TO STAY IN SIM.
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C              - IF =0; SITE IS NOT RADIATING
C              - IF =1; SITE IS RADIATING
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C     XS(JM) - X POSITION OF SITE
C     YS(JM) - Y POSITION OF SITE
C.....LOCAL TO THIS SUBROUTINE.....
C     ACTION - INTEGER INDICATING ACTION DESIRED IN MESSAGE
C     CTIME - CURRENT TIME: TIME THAT MESSAGE IS TO BE SENT
C     DELAY - EXTRA DELAY TIME DUE TO DAMAGE OF EW SITE
C     II - LOOP COUNTER
C     JJ - ARRAY POINTER
C     L  - EW SITE UNIT # FOR SAM SITE UNIT # OF INTEREST
C        - OR NON-SAM TYPE TARGET NUMBER (DEPENDS UPON USE)
C     INID - ID # OF INCOMING MESSAGE
C     POS - POSITION OF NEXT OPEN SLOT IN MESSAGE ID HOLDING ARRAY
C     RND1 - RANDOM NUMBER USED IN TICK2 SUB TO DETERMINE SURVIVAL
C          - OUTCOME IN A PROBABILISTIC WAY
C     UPDATE(IM) - HOLDS UPDATED DETECTION INFO FOR A RECENTLY KILLED
C                - EW SITE, USED IN UPDATE MESSAGE TO ADWOC CENTER
C.....START PROGRAM.....
```

```
C     WRITE(*,*) 'IADSPS'

      IF (TYPE .EQ. 1  .AND. EWSITE(UNIT1) .EQ. 1) THEN

C ...  TARGET HIT WAS AN EW SITE IN THE IADS....

C ...DETERMINE THE UNIT NUMBER FOR THIS EW SITE

        L=0
        DO 17 JJ=1,NEWSITES
        IF(EWID(JJ) .EQ. UNIT1) THEN
          L=JJ
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) '    SITE IS EW SITE #',L
          ENDIF
          GOTO 18
        ENDIF
   17   CONTINUE
   18   CONTINUE

        IF (L .EQ. 0) THEN
          WRITE(16,*) ' UNABLE TO DETERMINE EW SITE #, STOPPING'
          STOP
        ENDIF

        IF (PS(UNIT1) .LT. EWPSMIN) THEN
C ... EW SITE DEAD. REMOVE DETECTION DATA FROM ADWOC .....
C ... AND UPDATE TIME OF ANY MESSAGES FROM THIS SITE .....

          IF (DEBUG .GE. 1) THEN
            WRITE (16,1000) L
 1000         FORMAT('    DESTROYED SITE WAS EW SITE #',I2)
          ENDIF
C... DELETE ALL MESSAGE FROM THIS EW UNIT THAT HAVE NOT YET BEEN SENT
          IADSTYPE=1
          IADSUNIT=L

          CALL DELMES(IADSTYPE,IADSUNIT)

C ... SEND A MESSAGE TO THE IADS ELEMEMTS TO REMOVE DETECTION INFO
C ... SET ALL DETECTED TARGETS TO UNDETECTED STATUS .....
C ... FIRST GET AN UPDATED DETECTION REPORT FOR THE EW SITE ..

        CALL EWDET2(IADSUNIT,UPDATE)

        DO 110 II=1,NAC
        UPEWTGT(IADSUNIT,II)=-UPDATE(II)
        EWDETECT(IADSUNIT,II)=0
  110    CONTINUE
```

```
      DELAY=10.0

      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) '   CHANGE IN DETECTION STATUS FOR EW SITE #',
     &    IADSUNIT
        WRITE(16,*) '   (-1= LOST, 0= NO CHANGE, 1= NEW DETECTION)'
        WRITE(16,*) '   (1-NAC) UPDATE DETECTION ARRAY :',
     &    (UPEWTGT(IADSUNIT,LL),LL=1,NAC)
        WRITE(16,*) '   DELAY SET (SITE DEAD) TO=',DELAY,
     &    '  TIME WHEN MESSAGE WILL BE SENT=',TIME+DELAY
      ENDIF
      IF (DEBUG .GE. 1) THEN
        WRITE(16,1009) TIME,IADSUNIT,TIME+DELAY
1009     FORMAT(F5.0,' SEC EW SITE ',I2,' WILL SEND MESSAGE',
     &    ' AT TIME ',F5.0)
      ENDIF

      CALL EWMES(IADSUNIT,(TIME+DELAY))

C ... SEND A MESSAGE TO THE IADS ELEMEMTS TO REMOVE DETECTION INFO

      ELSEIF (PS(UNIT1) .GE. EWPSMIN) THEN
        IF (DEBUG .GE. 2) THEN
          WRITE(16,*) '   EW SITE INPUT DELAY PARAMETERS:'
          WRITE(16,*) EWPROCDLY(L),EWDAMDLY(L),EWDAMRAT(L),PS(J)
        ENDIF
        CALL CDELAY(EWPROCDLY(L),EWDAMDLY(L),EWDAMRAT(L),
     &    PS(J),DELAY)
        IF (DEBUG .GE. 2) THEN
          WRITE(16,*) ' COMPUTED DELAY DUE TO DAMAGE =',DELAY
        ENDIF
        IADSTYPE=1
        IADSUNIT=L
        CALL DLYMES(IADSTYPE,IADSUNIT,DELAY)
        IF (DEBUG .GE. 1) THEN
          WRITE (16,1010) L
1010       FORMAT('    IMPACTED SITE IS EW SITE # ',I2)
        ENDIF
      ENDIF
    ENDIF

    IF (TYPE .EQ. 1  .AND. EWSITE(UNIT1) .EQ. 0) THEN

C ...  TARGET HIT WAS A SAM SITE IN THE IADS....
      IADSTYPE=5
      IADSUNIT=UNIT1

      IF (PS(J) .LT. PSSMIN) THEN
        IF (DEBUG .GE. 1) THEN
          WRITE (16,1007) TIME,I,J,PS(J),RD1
```

```
1007        FORMAT(F5.0,' SEC STANDOFF WEAPON ',I2,' DESTROYS ',
     &      'SAM SITE #',I2,' PS=',F4.2,' RND#',F4.2)
          ENDIF
          CALL DELMES(IADSTYPE,IADSUNIT)
        ELSEIF (PS(J) .GE. PSSMIN) THEN
          IF (DEBUG .GE. 1) THEN
            WRITE (16,1008) TIME,I,J,PS(J),RD1
1008        FORMAT(F5.0,' SEC STANDOFF WEAPON ',I2,' IMPACTS',
     &      ' SAM SITE #',I2,' PS=',F4.2,' RND#',F4.2)
          ENDIF
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) '  SAM SITE INPUT DELAY PARAMETERS:'
            WRITE(16,*) SAMPROCDLY,SAMDAMDLY,SAMDAMRAT,PS(J)
          ENDIF
          CALL CDELAY(SAMPROCDLY,SAMDAMDLY,SAMDAMRAT,PS(J),DELAY)
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) ' COMPUTED DELAY DUE TO DAMAGE =',DELAY
          ENDIF
          CALL DLYMES(IADSTYPE,IADSUNIT,DELAY)
        ENDIF
      ENDIF

      IF (TYPE .EQ. 2) THEN

C ... THIS IS A NON-EW OF SAM SITE TYPE IADS TARGET  .....
C ... FIND OUT WHAT TYPE AND UNIT # THIS IADS ELEMENT IT IS..

        IADSTYPE=0
        IADSUNIT=0

        DO 30 II=1,NUMIADSTGTS
        IF(IDIADSTGT(II,1) .EQ. UNIT2) THEN
          IADSTYPE=IDIADSTGT(II,2)
          IADSUNIT=IDIADSTGT(II,3)
        ENDIF
   30   CONTINUE

        IF (IADSTYPE .EQ. 0  .OR. IADSUNIT .EQ. 0) THEN
          WRITE(16,*) ' UNABLE TO DETERMINE IADS TYPE OR UNIT #'
          WRITE(16,*) ' STOPPING SIMULATION IN SUB IADSPS'
          STOP
        ENDIF

       L=UNIT2

C ... DETERMINE NEW DELAY TIME AND THEN LOCATE MESSAGE(S) AND ALTER

        IF(IADSTYPE .EQ. 2) THEN
          IF (PST(L) .LT. MINRPTPS) THEN
            IF (DEBUG .GE. 1) THEN
```

```
            WRITE (16,1001) IADSUNIT
1001          FORMAT(5X,'NON-SAM TYPE TARGET IS RPT STA #',I2)
            ENDIF
            CALL DELMES(IADSTYPE,IADSUNIT)
          ELSEIF (PST(L) .GE. MINRPTPS) THEN
            IF (DEBUG .GE. 1) THEN
              WRITE (16,1002) TIME,I,IADSUNIT,PST(L),RD1
1002            FORMAT(F5.0,' SEC STANDOFF WEAPON ',I2,' IMPACTS',
     &          ' RPT STA # ',I2,' PS=',F4.2,' RND#',F4.2)
            ENDIF
            IF (DEBUG .GE. 2) THEN
              WRITE(16,*) '   RPT STA INPUT DELAY PARAMETERS:'
              WRITE(16,*) RPTPROCDLY(IADSUNIT),RPTDAMDLY(IADSUNIT),
     &          RPTDAMRAT(IADSUNIT),PST(L)
            ENDIF
            CALL CDELAY(RPTPROCDLY(IADSUNIT),RPTDAMDLY(IADSUNIT),
     &        RPTDAMRAT(IADSUNIT),PST(L),DELAY)
            IF (DEBUG .GE. 2) THEN
              WRITE(16,*) ' COMPUTED DELAY DUE TO DAMAGE =',DELAY
            ENDIF
            CALL DLYMES(IADSTYPE,IADSUNIT,DELAY)
          ENDIF
        ELSEIF(IADSTYPE .EQ. 3) THEN
          IF (PST(L) .LT. MINADWOCPS) THEN
            IF (DEBUG .GE. 1) THEN
              WRITE (16,1003) TIME,I,PST(L),RD1
1003            FORMAT(F5.0,' SEC STANDOFF WEAPON ',I2,' DESTROYS ',
     &          'ADWOC CENTER; PS=',F4.2,' RND#',F4.2)
            ENDIF
            CALL DELMES(IADSTYPE,IADSUNIT)
          ELSEIF (PST(L) .GE. MINADWOCPS) THEN
            IF (DEBUG .GE. 1) THEN
              WRITE (16,1004) TIME,I,PST(L),RD1
1004            FORMAT(F5.0,' SEC STANDOFF WEAPON ',I2,' IMPACTS',
     &          ' ADWOC CENTER; PS=',F4.2,' RND#',F4.2)
            ENDIF
            IF (DEBUG .GE. 2) THEN
              WRITE(16,*) '   ADWOC INPUT DELAY PARAMETERS:'
              WRITE(16,*) ADWPROCDLY,ADWDAMDLY,ADWDAMRAT,PST(L)
            ENDIF
            CALL CDELAY(ADWPROCDLY,ADWDAMDLY,ADWDAMRAT,PST(L),DELAY)
            IF (DEBUG .GE. 2) THEN
              WRITE(16,*) ' COMPUTED DELAY DUE TO DAMAGE =',DELAY
            ENDIF
            CALL DLYMES(IADSTYPE,IADSUNIT,DELAY)
          ENDIF
        ELSEIF(IADSTYPE .EQ. 4) THEN
          IF (PST(L) .LT. MINBATPS) THEN
            IF (DEBUG .GE. 1) THEN
              WRITE (16,1005) TIME,I,IADSUNIT,PST(L),RD1
```

```
1005          FORMAT(F5.0,' SEC STANDOFF WEAPON ',I2,' DESTROYS ',
     &        'BAT CNRT# ',I2,' PS=',F4.2,' RND#',F4.2)
            ENDIF
            CALL DELMES(IADSTYPE,IADSUNIT)
          ELSEIF (PST(L) .GE. MINBATPS) THEN
            IF (DEBUG .GE. 1) THEN
              WRITE (16,1006) TIME,I,IADSUNIT,PST(L),RD1
1006          FORMAT(F5.0,' SEC STANDOFF WEAPON ',I2,' IMPACTS',
     &        ' BAT CNRT# ',I2,' PS=',F4.2,' RND#',F4.2)
            ENDIF
            IF (DEBUG .GE. 2) THEN
              WRITE(16,*) '  BAT CNTR INPUT DELAY PARAMETERS:'
              WRITE(16,*) BATPROCDLY(IADSUNIT),BATDAMDLY(IADSUNIT),
     &        BATDAMRAT(IADSUNIT),PST(L)
            ENDIF
            CALL CDELAY(BATPROCDLY(IADSUNIT),BATDAMDLY(IADSUNIT),
     &       BATDAMRAT(IADSUNIT),PST(L),DELAY)
            IF (DEBUG .GE. 2) THEN
              WRITE(16,*) ' COMPUTED DELAY DUE TO DAMAGE =',DELAY
            ENDIF
            CALL DLYMES(IADSTYPE,IADSUNIT,DELAY)
          ENDIF
        ENDIF
      ENDIF

      RETURN
      END
-----------------------------------
      SUBROUTINE INITEW

C.....LATEST CHANGE MADE 6/28/91 BFS.....
C     SETS INITIAL STATUS EACH EW SITE AT THE START OF THE SIMULAITON.
C     STATUS =0; NON-RADIADITING. =1; RADIATING AND TRYING TO DETECT
C     PENETRATORS
C
C     CALLED BY - IADS
C     CALLS     - HOLDMES
C.....

      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER II
      REAL MSG(13)

C.....DECLARE VARIABLES IN COMMON BLOCK.....
```

```
      INTEGER DEBUG,STATS

C.....COMMON BLOCK.....

      INCLUDE 'QIADS DEF'
      INCLUDE 'EW1 DEF'
      INCLUDE 'MESID DEF'

      COMMON/DEBUG/DEBUG
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....VARIABLES IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C               - IF =0; SITE IS NOT RADIATING
C               - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C               - AND GETTING READY TO FIRE, SITE IS RADIATING
C               - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C               - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C.....LOCAL TO THIS SUBROUTINE.....
C     MSG(13) - HOLDS DATA CONTAINING MESSAGE TO BE SENT
C     II
C.....START PROGRAM.....

C     WRITE(*,*) 'INITEW'

      MSG(1)=1.0
      MSG(2)=10.0
      MSG(3)=1.0
      MSG(4)=3.0
      MSG(5)=1.0
      MSG(8)=1.0
      MSG(10)=3.0
      MSG(11)=1.0
      MSG(12)=1.0

C... SET ALL EW SITES TO RADIATING, I.E. TRYING TO DETECT TARGETS.....

      DO 10 II=1,NEWSITES
C ... SEND TO EW SITE DIRECT.....
      MSG(6)=1.0
      MSG(7)=REAL(II)
```

```
      MSG(9)=REAL(MSGID)
      MSG(13)=REAL(II)
      IF (DEBUG .GE. 1) THEN
        WRITE(16,1003) MSG(1),MSG(2),NINT(MSG(7))
 1003   FORMAT(F5.0,' SEC ADWOC WILL SEND START UP MESSAGE AT TIME',
     &  F5.0,' TO EW SITE # ',I2)
        WRITE(16,*) '          MSGID=',NINT(MSG(9))
      ENDIF
      CALL HOLDMES(MSG)
C ... SEND COPY OF MESSAGE TO RPT STATIONS TOO.....
      MSG(6)=2.0
      MSG(7)=1.0
      IF (DEBUG .GE. 1) THEN
        WRITE(16,1004) MSG(1),MSG(2),NINT(MSG(13))
 1004   FORMAT(F5.0,' SEC ADWOC WILL SEND START UP MESSAGE AT TIME',
     &  F5.0,' TO EW SITE # ',I2)
        WRITE(16,*) '          VIA RPT STA #',NINT(MSG(7)),
     &    ' MSGID=',NINT(MSG(9))
      ENDIF
      CALL HOLDMES(MSG)
      MSG(7)=2.0
      IF (DEBUG .GE. 2) THEN
        WRITE(16,1004) MSG(1),MSG(2),NINT(MSG(13))
        WRITE(16,*) '          VIA RPT STA #',NINT(MSG(7)),
     &    ' MSGID=',NINT(MSG(9))
      ENDIF
      CALL HOLDMES(MSG)
      MSGID=MSGID+1
   10 CONTINUE

C ... GENERATE A DUPLICATE MESSAGE AS A TEST
C     MSGID=MSGID-2
C     MSG(2)=20.0
C     MSG(8)=0.0
C     DO 20 II=1,NEWSITES
C     MSG(7)=REAL(II)
C     MSG(9)=REAL(MSGID)
C     CALL HOLDMES(MSG)
C     MSGID=MSGID+1
C  20 CONTINUE

      RETURN
      END
---------------------------------
      SUBROUTINE MESRDY

C.....LATEST CHANGE MADE 07/11/91 BFS.....
C     DETERMINES IF A MESSAGE IS READY TO SEND, DETERMINES THE IADS
C     TYPE THE MESSAGE IS GOING TO AND CALLS THE APPROPRIATE SUB.
C
```

```
C     CALLED BY - IADS
C     CALLS     - REW,REWSTA,RADWOC,RBAT,RSAM
C.....................................................................

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER GOINGTO,II,JJ,MESID
      REAL MESTIME
C.....DECLARE VARIABLES IN COMMON BLOCK.....

      REAL TIME,DT
      INTEGER DEBUG

C.....COMMON BLOCK.....
      INCLUDE 'HOLDM1 DEF'

      COMMON/DEBUG/DEBUG
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                                .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     DT - TIME STEP SIM. CLOCK IS INCREMENTED BY
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C.....LOCAL TO THIS SUBROUTINE.....
C     GOINGTO - IADS UNIT # THAT MESSAGE IS GOING TO
C     II - LOOP COUNTER
C     JJ - LOOP COUNTER
C     MESID - HOLDS MESSAGE ID #
C     MESTIME - HOLDS TIME WHEN MESSAGE WILL BE SENT
C.....START PROGRAM.....

C     WRITE(*,*) 'MESRDY'

C ... CHECK IF ANY MESSAGES ARE READY TO BE SENT.....
C ... CONSTRUCT A LOOP THAT WILL ALLOW POINTING AND ENDING INDEX
C ... TO BE MODIFIED WHEN MESSAGES ARE TAKEN OFF ...

      II=1
```

```
 10 MESTIME=MESHOLD(II,2)
    GOINGTO=NINT(MESHOLD(II,7))
    MESID=NINT(MESHOLD(II,9))

    IF(DEBUG .GE. 2) THEN
      WRITE(16,*)'MESSAGE POINTER =',II,' MESSAGE TIME =',MESTIME,
   &   ' CURRENT TIME =',TIME
    ENDIF
C ... IF TIME EQUALS ZERO THEN WE ARE AT THE END OF THE ARRAY ....

    IF(NINT(MESHOLD(II,1)) .EQ. 0) THEN
      IF(DEBUG .GE. 2) THEN
        WRITE(16,*)'MESSAGE POINTER =',II,' MESHOLD(II,1)=0',
   &     ' END OF ARRAY'
      ENDIF
      GO TO 100
    ENDIF

C ... IS MESSAGE NOW READY TO BE SENT .....

    IF(MESTIME .GE. TIME .AND. MESTIME .LT. TIME+DT) THEN
      IF(DEBUG .GE. 2) THEN
        WRITE(16,*) TIME,' SEC MESSAGE FOUND READY TO BE SENT',
   &     '; MSGID # =',MESID
      ENDIF

C ... FIND WHO RECEIVES THIS MESSAGE THATS READY .....

      IF(NINT(MESHOLD(II,6)) .EQ. 1) THEN
        IF(DEBUG .GE. 2) THEN
          WRITE(16,*)'MESSAGE READY FOR EW SITE',GOINGTO
        ENDIF
        CALL REW(II)
      ELSEIF(NINT(MESHOLD(II,6)) .EQ. 2) THEN
        IF(DEBUG .GE. 2) THEN
        WRITE(16,*)'MESSAGE READY FOR EW REPORTING STATION',GOINGTO
        ENDIF
        CALL REWSTA(II)
      ELSEIF(NINT(MESHOLD(II,6)) .EQ. 3) THEN
        IF(DEBUG .GE. 2) THEN
          WRITE(16,*)'MESSAGE READY FOR ADWOC CENTER'
        ENDIF
        CALL RADWOC(II)
      ELSEIF(NINT(MESHOLD(II,6)) .EQ. 4) THEN
        IF(DEBUG .GE. 2) THEN
          WRITE(16,*)'MESSAGE READY FOR BATTALION CENTER',GOINGTO
        ENDIF
        CALL RBAT(II)
      ELSEIF(NINT(MESHOLD(II,6)) .EQ. 5) THEN
        IF(DEBUG .GE. 2) THEN
```

```
          WRITE(16,*)'MESSAGE READY FOR SAM SITE',GOINGTO
        ENDIF
        CALL RSAM(II)
      ENDIF
    ENDIF

C... INCREMENT POINTER, IF > 200 THEN AT END OF ARRAY AND STOP LOOP ..

    II=II+1
    IF(II .GT. 200) THEN
      GO TO 100
    ENDIF

    GO TO 10

100  CONTINUE

    RETURN
    END
---------------------------------
    SUBROUTINE NSPS(TYPE,UNIT,PSURVIV)

C.....LATEST CHANGE MADE 07/30/91 BFS.....
C     DETERMINES THE NON-SAM TYPE TARGET ID NUMBER FOR THE PASSED IADS
C   TYPE AND ELEMENT. THEN DETERMINES THE ELEMENTS CURRENT PROBABILITY
C     OF SURVIVAL
C
C     CALLED BY - REWSTA,RADWOC,RBAT
C     CALLS     - NONE
C.....

    PARAMETER (IM=30,JM=25,KM=20)

    IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

    INTEGER TYPE,UNIT
    REAL PSURVIV

C.....DECLARE LOCAL VARIABLES.....

    INTEGER II,IDNUM

C.....DECLARE VARIABLES IN COMMON BLOCK.....

    INTEGER DEBUG
    REAL PST

C.....COMMON BLOCK.....
```

```
      INCLUDE 'CON1    DEF'

      COMMON/DEBUG/DEBUG
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     PST(20) - SURVIVAL PROBABILITY OF NON-SAM(IADS) TYPE TARGET
C.....LOCAL TO THIS SUBROUTINE.....
C     II - LOOP COUNTER
C     PSURVIV - SURVIVAL PROBABILITY FOR THE GIVEN IADS TYPE AND UNIT #
C.....START PROGRAM.....

C     WRITE(*,*) 'NSID'

      IDNUM=0

      DO 10 II=1,NUMIADSTGTS
      IF(IDIADSTGT(II,2) .EQ. TYPE .AND.
     &   IDIADSTGT(II,3) .EQ. UNIT) THEN
        IDNUM=IDIADSTGT(II,1)
      ENDIF
   10 CONTINUE

      IF (IDNUM .EQ. 0) THEN
        WRITE(16,*) ' UNABLE TO DETERMINE NON-SAM TARGET # FOR ',
     &   'IADS TYPE AND UNIT#;',UNIT,TYPE
        WRITE(16,*) ' STOPPING SIMULATION IN SUB NSID'
        STOP
      ENDIF

      PSURVIV=PST(IDNUM)


      RETURN
      END
---------------------------------
      SUBROUTINE PSEW0

C.....LATEST CHANGE MADE 6/26/91 BFS.....
C     REMOVES EW SITES FROM SIMULATION BY MAKING THE INITIAL PS = 0.0.
```

```
C     USED IN NON-IADS MODE WHEN IADS DATA IS IN THE INPUT DATAFILE
C
C     CALLED BY - REDIAD
C     CALLS     - NONE
C.....

      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER II

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      REAL PS
C.....COMMON BLOCK.....

      INCLUDE 'QIADS DEF'
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)


C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                                  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....IN COMMON BLOCK.....
C     EWID(9)  - ID # OF SAM SITES DESIGNATED AS EW SITES
C     NEWSITES - MAX # OF EW SITES IN SAM DATA BLOCK
C.....LOCAL TO THIS SUBROUTINE.....
C     II
C.....START PROGRAM.....

C     WRITE(*,*) 'PSEW0'

C ... SETS ALL SITES DESIGNATED AS EW TO ZERO TAKING THEM OUT OF SIM.
C ... USED WHEN EW SITES ARE IN SAM DATA BLOCK BUT RUNNING NON-IADS

C     WRITE(*,*) 'BEFORE ZEROING, PS OF SITES IS'
C     DO 20 II=1,3
C       WRITE(*,*) 'II,PS(II)',II,PS(II)
C  20 CONTINUE

      DO 10 II=1,NEWSITES
        PS(EWID(II))=0.0
   10 CONTINUE
```

```
C     WRITE(*,*) 'AFTER ZEROING, PS OF SITES IS'
C     DO 30 II=1,3
C       WRITE(*,*) 'II,PS(II)',II,PS(II)
C  30 CONTINUE

      RETURN
      END
---------------------------------
      SUBROUTINE RADWOC(II)

C.....LATEST CHANGE MADE 07/19/91 BFS.....
C     FOR MESSAGES GOING TO ADWOC CENTER. CALLS A SUB. THAT RECEIVES
C     A MESSAGE OFF HOLDING MESSAGE ARRAY AND PROCESSES IT. THE ADWOC
C     IS THE CENTER FOR DECISION MAKING BASED ON INFORMATION RECEIVED
C     FROM MESSAGES. IF THE INCOMING MESSAGES ID # IS UNIQUE, THE
C     MESSAGE AND INFORMATION IS RECEIVED. IF THE ID # IS NOT UNIQUE,
C     THE MESSAGE IS REJECTED, THUS AVOIDING REPETITIVE ACTIONS.
C
C     CALLED BY - MSGRDY
C     CALLS     - SHIFTU,HOLDMES,GETDET
C.....


      PARAMETER (IM=30,JM=25,KM=20)


      IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

      INTEGER II

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG

C.....COMMON BLOCK.....

      INCLUDE 'HOLDM1 DEF'
      INCLUDE 'ADWMES1 DEF'
      INCLUDE 'ADWDATA DEF'

      COMMON/DEBUG/DEBUG

C.....DECLARE LOCAL VARIABLES.....

      INTEGER JJ,MESID,ORIGTYPE,POS,PREVTYPE,ORIGUNIT,TYPE,UNIT
      REAL MSG(MAXMSGSIZE),PSURVIV

C..... VARIABLES USED IN THIS SUBROUTINE.....
```

```
C.....                                  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C.....LOCAL TO THIS SUBROUTINE.....
C     II - PASSED HOLDING MESSAGE ARRAY POINTER
C     JJ - LOOP COUNTER
C     MSG(MAXMSGSIZE) - HOLDS RECEIVED MESSAGE INFORMATION
C     MESID - INCOMING MESSAGE ID #
C     ORIGTYPE - INTEGER HOLDING THE ORIGIN SOURCE MESSAGE IADS TYPE #
C     ORIGUNIT - INTEGER HOLDING THE ORIGIN SOURCE MESSAGE IADS UNIT #
C     POS - POSITION OF NEXT OPEN SLOT IN MESSAGE ID HOLDING ARRAY
C     PREVTYPE - INTEGER HOLDING THE PREVIOUS SOURCE IADS TYPE #
C     PSURVIV - CURRENT SURVIVAL PROBABILITY OF THIS UNIT #
C     TYPE - IADS ADWOC CENTER TYPE NUMBER (=3)
C     UNIT - HOLDS UNIT #(=1)
C.....START PROGRAM.....

C     WRITE(*,*) 'RADWOC'

C ... RECEIVE DATA FROM MESSAGE .....

      PREVTYPE=NINT(MESHOLD(II,4))
      MESID=NINT(MESHOLD(II,9))
      ORIGTYPE=NINT(MESHOLD(II,10))
      ORIGUNIT=NINT(MESHOLD(II,11))


      TYPE=3
      UNIT=NINT(MESHOLD(II,7))

C ... DETERMINE IF ADWOC IS ALIVE, RETURN OF DEAD.....

      CALL NSPS(TYPE,UNIT,PSURVIV)
      IF(PSURVIV .LT. MINADWOCPS) THEN
        IF(DEBUG .GE. 1) THEN
         WRITE(16,*)'   ADWOC DEAD; DOES NOT RECEIVE ANY',
     &    ' MESSAGES OR SEND ANY'
         WRITE(16,*)'   PSURVIVAL=',PSURVIV,' MIN PS=',MINADWOCPS
        ENDIF
        RETURN
      ENDIF

C ... CONTINUE IF ADWOC IS NOT DEAD.....


      IF(DEBUG .GE. 2) THEN
```

```
      WRITE(16,*)'   IN RADWOC, RECEIVING MES ID# =',MESID
    ENDIF

C ... COMPARE INCOMING MESSAGE ID WITH ID # OF PREVIOUSLY RECEIVED
C ... MESSAGES, IF A MATCH; IGNORE MESSAGE, TAKE IT OFF HOLDING ARRAY

    POS=ADWMESIDPOS
    DO 10 JJ=1,POS-1
    IF(ADWMESID(JJ) .EQ. MESID) THEN
      IF(DEBUG .GE. 2) THEN
        WRITE(16,*)'      MESSAGE #',MESID,' IGNORED;',
   &      ' MATCHES WITH PREVIOUS ID#'
      ENDIF
      CALL SHIFTU(II)
      RETURN
    ENDIF
 10 CONTINUE

C ... TO GET HERE, THIS IS NEW MESSAGE SO HOLD ITS ID # .....

C ... CHECK THAT YOU HAVE ROOM TO ADD ANOTHER ID # TO ID ARRAY.....

    IF(POS .GT. MAXADWMESID) THEN
      WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
      WRITE(*,*)  'ADWOC NEEDS TO STORE NEW ID # AND',
   &   ' ADWOC ID ARRAY IS FULL'
      WRITE(*,*)  'PRESENT VALUE OF MAXADWMESID PARAMETER IS',POS-1
      WRITE(*,*)  'INCREASE MAXADWMESID IN ADWMES1 DEF AND',
   &   ' RECOMPILE PROGRAM MODULES'
      WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
      WRITE(16,*) ' ***    PROGRAM STOPPED    ***'
      WRITE(16,*)  'ADWOC NEEDS TO STORE NEW ID # AND',
   &   ' ADWOC ID ARRAY IS FULL'
      WRITE(16,*)  'PRESENT VALUE OF MAXADWMESID PARAMETER IS',POS-1
      WRITE(16,*)  'INCREASE MAXADWMESID IN ADWMES1 DEF AND',
   &   ' RECOMPILE PROGRAM MODULES'
      WRITE(16,*) ' ***    PROGRAM STOPPED    ***'
    STOP
    ENDIF

C ... ARRAY HAS ROOM SO PUT NEW MSG ID # INTO IT AND INCREMENT POINTER

    ADWMESID(POS)=MESID
    ADWMESIDPOS=ADWMESIDPOS+1

C.....TRANSFER MESSAGE INTO LOCAL HOLDING ARRAY.....

    DO 15 JJ=1,MAXMSGSIZE
    MSG(JJ)=MESHOLD(II,JJ)
 15 CONTINUE
```

```
C.....DELETE CURRENT MESSAGE FROM HOLDING MESSAGE ARRAY .....

      CALL SHIFTU(II)



      IF(PREVTYPE .EQ. 1) THEN

C ... MESSAGE IS FROM AN EW SITE, RECEIVE THE MESSAGE AND ASSOCIATED
C ... DETECTION DATA.....

      CALL GETDET(ORIGUNIT,MESID,MSG(2))

      ELSEIF(PREVTYPE .EQ. 2) THEN

C ... MESSAGE COMES FROM AN EW REPORTING STATION.

      IF(ORIGTYPE .EQ. 1) THEN
C.... MESSAGE IS FROM AN EW SITE. DO THE SAME AS IF COMING FROM EW SITE
        CALL GETDET(ORIGUNIT,MESID,MSG(2))
      ENDIF

      ELSEIF(PREVTYPE .EQ. 4) THEN
C ... SOURCE OF MESSAGE IS A BATTALION CENTER

      ELSEIF(PREVTYPE .EQ. 5) THEN
C ... SOURCE OF MESSAGE IS A SAM UNIT
      ENDIF

      RETURN
      END
--------------------------------
      SUBROUTINE RBAT(II)

C.....LATEST CHANGE MADE 07/24/91 BFS.....
C     FOR MESSAGES GOING TO SAM SITES ONLY, THIS SUB.
C     RECEIVES MESSAGE OFF HOLDING MESSAGE ARRAY AND PROCESSES IT.
C     BATTALION STATIONS ARE MESSAGES PASSER ONLY AND DO NO ACTION
C     OF THEMSELVES. THE SOURCE AND DESTINATION OF THE MESSAGE WILL
C     DETERMINE WHERE COPIES OF THE MESSAGE GET SENT. IF THE MESSAGES
C     ID # IS UNIQUE THE MESSAGE IS PASSED ALONG. IF IT IS NOT UNIQUE,
C     THE MESSAGE IS NOT PASSED ALONG, THUS AVOIDING INFINITE LOOPS AND
C     MULTIPLE MESSAGES.
C
C     CALLED BY - MSGRDY
C     CALLS     - SHIFTU,CDELAY,HOLDMES
C.....

      PARAMETER (IM=30,JM=25,KM=20)
```

```
      IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

      INTEGER II

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG
      REAL TIME,PST
C.....COMMON BLOCK.....

      INCLUDE 'QIADS DEF'
      INCLUDE 'EW1 DEF'
      INCLUDE 'HOLDM1 DEF'
      INCLUDE 'MESID DEF'
      INCLUDE 'CON1 DEF'
      INCLUDE 'BATMES1 DEF'
      INCLUDE 'BATDATA DEF'
      INCLUDE 'ADWDATA DEF'

      COMMON/DEBUG/DEBUG
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER ACTION,JJ,L,LL,MESID,POS,PREVTYPE,UNIT,TYPE,
     & DESTTYPE,DESTUNIT
      REAL MSG(MAXMSGSIZE),DELAY,PSURVIV

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                                .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C.....LOCAL TO THIS SUBROUTINE.....
C     ACTION - INTEGER INDICATING ACTION DESIRED IN MESSAGE
C     DELAY - TIME DELAY FROM PRESENT TIME WHEN MESSAGE WILL BE SENT
C     DESTTYPE - INTEGER HOLDING DESTINATION SOURCE MESSAGE IADS TYPE #
C     DESTUNIT - INTEGER HOLDING DESTINATION SOURCE MESSAGE IADS UNIT #
C     II - PASSED HOLDING MESSAGE ARRAY POINTER
C     L - LOOP COUNTER
C     LL - LOOP COUNTER
```

```
C     MSG(MAXMSGSIZE) - HOLDS RECEIVED MESSAGE INFORMATION
C     MESID - INCOMING MESSAGE ID #
C     POS - POSITION OF NEXT OPEN SLOT IN MESSAGE ID HOLDING ARRAY
C     PREVTYPE - INTEGER HOLDING THE PREVIOUS SOURCE IADS TYPE #
C     PSURVIV - CURRENT SURVIVAL PROBABILITY OF THIS UNIT #
C     TYPE - IADS EW REPORTING TYPE NUMBER (=2)
C     UNIT - HOLDS UNIT #
C.....START PROGRAM.....

C     WRITE(*,*) 'RBAT'

C ... RECEIVE DATA FROM MESSAGE .....

      TYPE=4
      UNIT=NINT(MESHOLD(II,7))
      ACTION=NINT(MESHOLD(II,8))
      PREVTYPE=NINT(MESHOLD(II,4))
      MESID=NINT(MESHOLD(II,9))
      DESTTYPE=NINT(MESHOLD(II,12))
      DESTUNIT=NINT(MESHOLD(II,13))

C ... DETERMINE IF THIS BATTALION CENTER IS ALIVE.....

      CALL NSPS(TYPE,UNIT,PSURVIV)
      IF(PSURVIV .LT. MINBATPS) THEN
        IF(DEBUG .GE. 1) THEN
         WRITE(16,*)'   BAT STA',UNIT,'DEAD; DOES NOT RECEIVE ANY',
     &    ' MESSAGES OR SEND ANY'
         WRITE(16,*)'   PSURVIVAL=',PSURVIV,' MIN PS=',MINBATPS
        ENDIF
        RETURN
      ENDIF

C ... CONTINUE IF BATTALION CENTE IS NOT DEAD .....


      IF(DEBUG .GE. 2) THEN
        WRITE(16,*)'   IN RBAT, BAT CTR',UNIT,'RECEIVES ',
     &   'MES ID#=',MESID
      ENDIF

C ... COMPARE INCOMING MESSAGE ID WITH ID # OF PREVIOUSLY RECEIVED
C ... MESSAGES, IF A MATCH; IGNORE MESSAGE, TAKE IT OFF HOLDING ARRAY

      POS=BATMESIDPOS(UNIT)
      DO 10 JJ=1,POS-1
      IF(BATMESID(UNIT,JJ) .EQ. MESID) THEN
        IF(DEBUG .GE. 2) THEN
          WRITE(16,*)'      MESSAGE ID# ',MESID,' IGNORED;',
     &       ' MATCHES WITH PREVIOUS ID#'
```

```
        ENDIF
        CALL SHIFTU(II)
        RETURN
      ENDIF
   10 CONTINUE

C ... TO GET HERE, THIS IS NEW MESSAGE SO HOLD ITS ID # .....

C ... CHECK THAT YOU HAVE ROOM TO ADD ANOTHER ID # TO ID ARRAY.....

      IF(POS .GT. MAXBATMESID) THEN
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(*,*)  'BAT STA #',UNIT,' NEEDS TO STORE NEW ID # AND',
     &   ' BAT ID ARRAY IS FULL'
        WRITE(*,*)  'PRESENT VALUE OF MAXBATMESID PARAMETER IS',POS-1
        WRITE(*,*)  'INCREASE MAXBATMESID IN BATMES1 DEF AND',
     &   ' RECOMPILE PROGRAM MODULES'
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(16,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(16,*) 'BAT STA #',UNIT,' NEEDS TO STORE NEW ID # AND',
     &   ' BAT ID ARRAY IS FULL'
        WRITE(16,*) 'PRESENT VALUE OF MAXBATMESID PARAMETER IS',POS-1
        WRITE(16,*) 'INCREASE MAXBATMESID IN BATMES1 DEF AND',
     &   ' RECOMPILE PROGRAM MODULES'
        WRITE(16,*) ' ***    PROGRAM STOPPED    ***'
       STOP
       ENDIF

C ... ARRAY HAS ROOM SO BUT NEW MSG ID # INTO IT AND INCREMENT POINTER

       BATMESID(UNIT,POS)=MESID
       BATMESIDPOS(UNIT)=BATMESIDPOS(UNIT)+1

C.....TRANSFER DATA INTO LOCAL HOLDING ARRAY.....

       DO 15 JJ=1,MAXMSGSIZE
       MSG(JJ)=MESHOLD(II,JJ)
    15 CONTINUE
       IF(DEBUG .EQ. 2) THEN
         WRITE(16,*) '   RECEIVED BAT MES.=',(MSG(JJ),JJ=1,MAXMSGSIZE)
       ENDIF

C.....DELETE CURRENT FROM HOLDING MESSAGE ARRAY ..... .

       CALL SHIFTU(II)

C ...DETERMINE THE ARRAY INDEX FOR THIS IADS ELEMENT IN THE NON-SAM
C ... TARGET PS ARRAY .....

       DO 17 JJ=1,NUMIADSTGTS
```

```
      IF(IDIADSTGT(JJ,2) .EQ. 4 .AND. IDIADSTGT(JJ,3) .EQ. UNIT) THEN
        L=IDIADSTGT(JJ,1)
        IF (DEBUG .GE. 2) THEN
          WRITE(16,*) '  BATTALION CENTER #',UNIT,
     &    ' IS NON-SAM TARGET # =',L
        ENDIF
        GOTO 18
      ENDIF
   17 CONTINUE
   18 CONTINUE

C ... COMPUTE TIME DELAY OF MESSAGE .....

      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) '  INPUTED BATTALION CENTER DELAY PARAMETERS:'
        WRITE(16,*) BATPROCDLY(UNIT),BATDAMDLY(UNIT),
     &  BATDAMRAT(UNIT),PST(L)
      ENDIF
      CALL CDELAY(BATPROCDLY(UNIT),BATDAMDLY(UNIT),BATDAMRAT(UNIT),
     &  PST(L),DELAY)
      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) '  COMPUTED BAT. CENTER DELAY=',DELAY,
     &    ' TIME WHEN MESSAGE WILL BE SENT=',TIME+DELAY
      ENDIF
      IF (DEBUG .GE. 1) THEN
        WRITE(16,1003) TIME,UNIT,TIME+DELAY,MESID
 1003   FORMAT(F5.0,' SEC BAT CNTR ',I2,' WILL SEND MESSAGE',
     &    ' AT TIME ',F5.0,' MSGID # ',I2)
      ENDIF
      MSG(2)=TIME+DELAY

C.....DETERMINE SOURCE OF MESSAGE AND APPROPRIATE SENDING DESTINATION .

      IF(PREVTYPE .EQ. 1) THEN

C ... MESSAGE IS FROM AN EW SITE, NO CONNECTION PRESENTLY ASSUMED

      ELSEIF(PREVTYPE .EQ. 2) THEN

C.... MESSAGE FROM AN EW REPORTING STATION IS AN ALERT.
C ... DIRECT ALERT TO THE SAM SITES AND SEND A COPY TO
C.....OTHER BATTALION CENTERS.....


C... PASS COPY OF MESSAGE TO SAM SITES I.E. CONNECT(L,3)=5 .....

        DO 30 L=1,NUMCON
        IF(CONNECT(L,1) .EQ. 4  .AND. CONNECT(L,2) .EQ. UNIT) THEN
          IF(CONNECT(L,3) .EQ. 5) THEN
            MSG(4)=REAL(CONNECT(L,1))
```

```
            MSG(5)=REAL(CONNECT(L,2))
            MSG(6)=REAL(CONNECT(L,3))
            MSG(7)=REAL(CONNECT(L,4))
            IF (DEBUG .GE. 2) THEN
            WRITE(16,*) 'BAT CNTR #',NINT(MSG(5)),' WILL SEND MES.',
     &        ' TO SAM SITE #',NINT(MSG(7)),'MSGID=',NINT(MSG(9))
            ENDIF
            IF (DEBUG .GE. 3) THEN
              WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &         (MSG(LL),LL=1,MAXMSGSIZE)
            ENDIF
            CALL HOLDMES(MSG)
          ENDIF
        ENDIF
   30   CONTINUE

C... PASS COPY OF MESSAGE TO OTHER BATTALION CENTERS

        DO 40 L=1,NUMCON
        IF(CONNECT(L,1) .EQ. 4  .AND. CONNECT(L,2) .EQ. UNIT) THEN
          IF(CONNECT(L,3) .EQ. 4) THEN
            MSG(4)=REAL(CONNECT(L,1))
            MSG(5)=REAL(CONNECT(L,2))
            MSG(6)=REAL(CONNECT(L,3))
            MSG(7)=REAL(CONNECT(L,4))
            IF (DEBUG .GE. 2) THEN
            WRITE(16,*) 'BAT CNTR #',NINT(MSG(5)),' WILL SEND MES.',
     &        ' TO BAT CNTR #',NINT(MSG(7)),'MSGID=',NINT(MSG(9))
            ENDIF
            IF (DEBUG .GE. 3) THEN
              WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &         (MSG(LL),LL=1,MAXMSGSIZE)
            ENDIF
            CALL HOLDMES(MSG)
          ENDIF
        ENDIF
   40   CONTINUE

      ELSEIF(PREVTYPE .EQ. 3) THEN

C.... MESSAGE IS FROM ADWOC CENTER. PASS MESSAGE ALONG TO SAM SITE
C ... DESIGNATED AND A COPY TO OTHER BATTALION CENTERS

C ... SEND MESSAGE TO THE DESIGNATED SAM SITE ONLY .....
        MSG(4)=REAL(CONNECT(L,1))
        MSG(5)=REAL(CONNECT(L,2))
        MSG(6)=MSG(12)
        MSG(7)=MSG(13)
        IF (DEBUG .GE. 2) THEN
         WRITE(16,*) 'BAT CNTR #',NINT(MSG(5)),' WILL SEND MESSAGE',
```

```
&    ' TO SAM SITE #',NINT(MSG(7)),'MSGID=',NINT(MSG(9))
     ENDIF
     IF (DEBUG .GE. 3) THEN
       WRITE(16,*) '   MESSAGE TO BE SEND IS:',
&      (MSG(LL),LL=1,MAXMSGSIZE)
     ENDIF
     CALL HOLDMES(MSG)

C... PASS COPY OF MESSAGE TO OTHER BATTALION CENTERS

     DO 50 L=1,NUMCON
     IF(CONNECT(L,1) .EQ. 4  .AND. CONNECT(L,2) .EQ. UNIT) THEN
       IF(CONNECT(L,3) .EQ. 4) THEN
         MSG(4)=REAL(CONNECT(L,1))
         MSG(5)=REAL(CONNECT(L,2))
         MSG(6)=REAL(CONNECT(L,3))
         MSG(7)=REAL(CONNECT(L,4))
         IF (DEBUG .GE. 2) THEN
         WRITE(16,*) 'BAT CNTR #',NINT(MSG(5)),' WILL SEND MES.',
&          ' TO BAT CNTR #',NINT(MSG(7)),'MSGID=',NINT(MSG(9))
         ENDIF
         IF (DEBUG .GE. 3) THEN
           WRITE(16,*) '   MESSAGE TO BE SEND IS:',
&           (MSG(LL),LL=1,MAXMSGSIZE)
         ENDIF
         CALL HOLDMES(MSG)
       ENDIF
     ENDIF
50   CONTINUE


    ELSEIF(PREVTYPE .EQ. 4) THEN

C ... SOURCE OF MESSAGE IS A BATTALION CENTER

     IF(DESTTYPE .EQ. 3) THEN

C ... DESTINATION OF MESSAGE IS ADWOC CENTER

C ... SEND MESSAGE TO THE ADWOC CENTER.....

       MSG(4)=4.0
       MSG(5)=REAL(UNIT)
       MSG(6)=MSG(12)
       MSG(7)=MSG(13)
       IF (DEBUG .GE. 2) THEN
         WRITE(16,*) 'BAT CNTR #',NINT(MSG(5)),' WILL SEND',
&         ' MESSAGE TO ADWOC CENTER; MSGID=',NINT(MSG(9))
       ENDIF
       IF (DEBUG .GE. 3) THEN
```

```
          WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &      (MSG(LL),LL=1,MAXMSGSIZE)
        ENDIF
        CALL HOLDMES(MSG)

      ELSEIF(DESTTYPE .EQ. 5) THEN

C ... DESTINATION OF MESSAGE IS SAM SITE
C ... SEND MESSAGE TO THE DESIGNATED SAM SITE ONLY .....

        MSG(4)=4.0
        MSG(5)=REAL(UNIT)
        MSG(6)=MSG(12)
        MSG(7)=MSG(13)
        IF (DEBUG .GE. 2) THEN
          WRITE(16,*) 'BAT CNTR #',NINT(MSG(5)),' WILL SEND MES.',
     &      ' TO SAM SITE #',NINT(MSG(7)),'MSGID=',NINT(MSG(9))
        ENDIF
        IF (DEBUG .GE. 3) THEN
          WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &      (MSG(LL),LL=1,MAXMSGSIZE)
        ENDIF
        CALL HOLDMES(MSG)

      ENDIF


     ELSEIF(PREVTYPE .EQ. 5) THEN

C ... SOURCE OF MESSAGE IS A SAM UNIT, SEND COPY OF MESSAGE TO ADWOC
C ... CENTER AND OTHER BATTALION CENTERS .....

C ... SEND MESSAGE TO THE ADWOC CENTER.....

        MSG(4)=4.0
        MSG(5)=REAL(UNIT)
        MSG(6)=MSG(12)
        MSG(7)=MSG(13)
        IF (DEBUG .GE. 2) THEN
          WRITE(16,*) 'BAT CNTR #',NINT(MSG(5)),' WILL SEND',
     &     ' MESSAGE TO ADWOC CENTER; MSGID=',NINT(MSG(9))
        ENDIF
        IF (DEBUG .GE. 3) THEN
          WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &     (MSG(LL),LL=1,MAXMSGSIZE)
        ENDIF
        CALL HOLDMES(MSG)

C... PASS COPY OF MESSAGE TO OTHER BATTALION CENTERS
```

```
      DO 60 L=1,NUMCON
      IF(CONNECT(L,1) .EQ. 4  .AND. CONNECT(L,2) .EQ. UNIT) THEN
        IF(CONNECT(L,3) .EQ. 4) THEN
          MSG(4)=REAL(CONNECT(L,1))
          MSG(5)=REAL(CONNECT(L,2))
          MSG(6)=REAL(CONNECT(L,3))
          MSG(7)=REAL(CONNECT(L,4))
          IF (DEBUG .GE. 2) THEN
          WRITE(16,*) 'BAT CNTR #',NINT(MSG(5)),' WILL SEND MES.',
     &       ' TO BAT CNTR #',NINT(MSG(7)),'MSGID=',NINT(MSG(9))
          ENDIF
          IF (DEBUG .GE. 3) THEN
            WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &        (MSG(LL),LL=1,MAXMSGSIZE)
          ENDIF
          CALL HOLDMES(MSG)
        ENDIF
      ENDIF
   60   CONTINUE

      ENDIF

      RETURN
      END
---------------------------------
      SUBROUTINE REDIAD

C.....LATEST CHANGE MADE 7/16/91 BFS.....
C     READS IN THE EXTRA DATA NEEDED BY IADS MODIFICATIONS. IF IN A
C     NON-IADS MODE, CALLS PSEW0. AFTER READING IN DATA, SETS THE
C     EWSITE(*) ARRAY WHICH INDICATES WHICH SAM SITES ARE EW SITES
C
C     CALLED BY - REDIN
C     CALLS     - PSEW0,SETID
C.....

      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER JJ,A,B,C

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER IOP

C.....COMMON BLOCK.....
```

```
      INCLUDE 'QIADS DEF'
      INCLUDE 'EWDATA DEF'
      INCLUDE 'RPTDATA DEF'
      INCLUDE 'ADWDATA DEF'
      INCLUDE 'BATDATA DEF'
      INCLUDE 'SAMDATA DEF'
      INCLUDE 'CON1 DEF'

      COMMON/CRUISE/IOP(10),TIMEL,RRAD(JM),IT(JM)

C.....DESCRIPTION OF VARIABLES IN THIS SUBROUTINE.....
C.....                                  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....VARIABLES IN COMMON BLOCK USED IN THIS SUBROUTINE .....
C   J - SAM SITE LOOP COUNTER
C
C.....VARIABLES LOCAL TO THIS SUBROUTINE.....
C     JJ
C     A
C     B
C     C
C.....START PROGRAM.....

C     WRITE(*,*) 'REDIAD'

C.....READ DESIGNATED EW SITE ID NUMBERS .....

      DO 250 JJ=1,NEWSITES
      READ(15,*) EWID(JJ)
  250 CONTINUE

C .....IF NOT IADS BUT EW SITES EXIST THEN SET PS TO ZERO ...

      IF (IOP(3) .LE. 4  .AND.  NEWSITES .GT. 0) THEN
        CALL PSEW0
        RETURN
      ENDIF

C.....READ EW SITE PARAMETERS .....

      DO 260 JJ=1,NEWSITES
      READ(15,*) EWPROCDLY(JJ),EWDAMDLY(JJ),EWDAMRAT(JJ)
  260 CONTINUE

      READ(15,*) EWPSMIN

C.....READ EW REPORTING SITE PARAMETERS .....
```

```
      READ(15,*) NUMRPT
      DO 261 JJ=1,NUMRPT
      READ(15,*) RPTPROCDLY(JJ),RPTDAMDLY(JJ),RPTDAMRAT(JJ)
  261 CONTINUE
      READ(15,*) MINRPTPS

C.....READ ADWOC CENTER PARAMETERS .....

      READ(15,*) ADWPROCDLY,ADWDAMDLY,ADWDAMRAT
      READ(15,*) MINADWOCPS,PSADWOC

C.....READ BATTALION CENTER PARAMETERS .....

      READ(15,*) NUMBAT
      DO 262 JJ=1,NUMBAT
      READ(15,*) BATPROCDLY(JJ),BATDAMDLY(JJ),BATDAMRAT(JJ)
  262 CONTINUE
      READ(15,*) MINBATPS

C.....READ SAM SITE PARAMETERS .....

      READ(15,*) SAMPROCDLY,SAMDAMDLY,SAMDAMRAT

C... READ IN IADS STRUCTURE .....
      READ(15,*) NUMCON
      DO 290 JJ=1,NUMCON
      READ(15,*) (CONNECT(JJ,L),L=1,4)
  290 CONTINUE

C... READ IN IADS ELEMENTS THAT WILL BE (NON-SAM TYPE) TARGETS .....

      READ(15,*) NUMIADSTGTS

      DO 300 JJ=1,NUMIADSTGTS
      READ(15,*) A,B,C
      IDIADSTGT(JJ,1)=A
      IDIADSTGT(JJ,2)=B
      IDIADSTGT(JJ,3)=C
  300 CONTINUE


C.....DONE READING IN IADS DATA .....

C ....SET =1 IN EWSITE(JM) FOR EVERY SAM SITE THAT REALLY IS EW SITE..
C ... OTHERWISE SET IT =0

      DO 270 JJ=1,JM
      EWSITE(JJ)=0
  270 CONTINUE
```

```
      DO 280 JJ=1,NEWSITES
      EWSITE(EWID(JJ))=1
  280 CONTINUE

      CALL SETID

      RETURN
      END
--------------------------------
      SUBROUTINE REW(II)

C.....LATEST CHANGE MADE 07/23/91 BFS.....
C     FOR MESSAGES GOING TO EW SITES ONLY, THIS SUBROUTINE
C     RECEIVES MESSAGE OFF HOLDING MESSAGE ARRAY AND PROCESSES IT.
C     STORES NEW MESSAGE ID# IN ARRAY. IF INCOMING MESSAGE ID HAS
C     ALREADY BEEN RECORDED, MESSAGE IS IGNORED.
C     PRESENTLY IT IS ASSUMED THAT THE ONLY MESSAGE SENT TO AN EW SITE
C     WILL BE TO TELL IT TO TURN ON OR OFF. EW SITES DO NOT SEND ANY
C     MESSAGES UPON THE RECEIPT OF A MESSAGE.
C
C     CALLED BY - MSGRDY
C     CALLS     - SHIFTU,EWMES,CDELAY
C.....

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

      INTEGER II

C.....DECLARE LOCAL VARIABLES.....

      INTEGER ACTION,EWUNIT,INID,J1,JJ,L,POS
      REAL CTIME,DELAY

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG,STATS,I,NAC

      REAL PS,TIME

C.....COMMON BLOCK.....

      INCLUDE 'QIADS DEF'
      INCLUDE 'EW1 DEF'
      INCLUDE 'EWDATA DEF'
      INCLUDE 'EWMES2 DEF'
      INCLUDE 'HOLDM1 DEF'
```

```
      COMMON/DEBUG/DEBUG
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     NAC - MAX # OF PENETRATORS
C     PS(JM) - SURVIVAL PROBABILITY OF SITE JM
C              - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C              - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C              - IF =0; SITE IS NOT RADIATING
C              - IF =1; SITE IS RADIATING
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C.....LOCAL TO THIS SUBROUTINE.....
C     ACTION - INTEGER INDICATING ACTION DESIRED IN MESSAGE
C     CTIME - CURRENT TIME: TIME THAT MESSAGE IS TO BE SENT
C     DELAY - EXTRA DELAY TIME DUE TO DAMAGE OF EW SITE
C     EWUNIT - HOLDS EW UNIT #
C     II - PASSED HOLDING MESSAGE ARRAY POINTER
C     J1 - ARRAY POINTER
C     JJ - ARRAY POINTER
C     L  - ARRAY POINTER
C     INID - ID # OF INCOMING MESSAGE
C     POS - POSITION OF NEXT OPEN SLOT IN MESSAGE ID HOLDING ARRAY
C.....START PROGRAM.....

C     WRITE(*,*) 'REW'

C ... RECEIVE DATA FROM MESSAGE AND ACT ON IT .....
C ... MESSAGES COME FROM ADWOC AND TELL EW SITE TO TURN ON OR OFF.
C ... DETERMINE THE EW UNIT # AND IMPLEMENT ACTION..

      EWUNIT=NINT(MESHOLD(II,7))
      ACTION=NINT(MESHOLD(II,8))
      INID=NINT(MESHOLD(II,9))
      CTIME=MESHOLD(II,2)
```

```
      IF(DEBUG .GE. 2) THEN
        WRITE(16,*)'   IN REW SUB, MES. REC. FOR EW SITE',EWUNIT,
     &   'ACTION =',ACTION,' MSG ID# =',INID
      ENDIF

C ... COMPARE INCOMING MESSAGE ID WITH ID # OF PREVIOUSLY RECEIVED
C ... MESSAGES, IF A MATCH IGNORE MESSAGE

      POS=REWMESIDPOS(EWUNIT)

      DO 10 JJ=1,POS-1
      IF(REWMESID(EWUNIT,JJ) .EQ. INID) THEN
        IF(DEBUG .GE. 2) THEN
          WRITE(16,*)'   MESSAGE ID#',INID,'IGNORED;',
     &      ' MATCHES WITH PREVIOUS ID#'
        ENDIF
        CALL SHIFTU(II)
        RETURN
      ENDIF
   10 CONTINUE

C ... TO GET HERE, THIS IS NEW MESSAGE SO HOLD ITS ID # AND PROCESS IT

C ... CHECK THAT YOU HAVE ROOM TO ADD ANOTHER ID # TO ID ARRAY.....

      IF(POS .GT. MAXEWMESID) THEN
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(*,*)  'EW SITE NEEDS TO STORE NEW ID # AND',
     &   ' EW SITE ID ARRAY IS FULL'
        WRITE(*,*)  'PRESENT VALUE OF MAXEWMESID PARAMETER IS',POS-1
        WRITE(*,*)  'INCREASE MAXEWMESID IN EWMES2 DEF AND',
     &   ' RECOMPILE PROGRAM MODULES'
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(16,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(16,*)  'EW SITE NEEDS TO STORE NEW ID # AND',
     &   ' EW SITE ID ARRAY IS FULL'
        WRITE(16,*)  'PRESENT VALUE OF MAXEWMESID PARAMETER IS',POS-1
        WRITE(16,*)  'INCREASE MAXEWMESID IN EWMES2 DEF AND',
     &   ' RECOMPILE PROGRAM MODULES'
        WRITE(16,*) ' ***    PROGRAM STOPPED    ***'
      STOP
      ENDIF

C ... ADD NEW MESSAGE ID # TO MESSAGE ID NUMBER HOLDING ARRAY .....

      REWMESID(EWUNIT,POS)=INID
      REWMESIDPOS(EWUNIT)=REWMESIDPOS(EWUNIT)+1


      IF(ACTION .EQ. 0) THEN
```

```
C ... ORDERED TO TURN OFF BUT IS ALREADY OFF BECAUSE IT IS DEAD.....

      IF (PS(EWID(EWUNIT)) .LT. EWPSMIN) THEN
        IF (DEBUG .GE. 1) THEN
         WRITE(16,*) '   EW SITE #,PS,EWPSMIN',
     &      EWUNIT,PS(EWID(EWUNIT)),EWPSMIN
         WRITE(16,*) '   ORDERED TO TURN OFF BUT ALREADY DEAD'
        ENDIF
        STATS(EWID(EWUNIT))=ACTION
        EWSTATS(EWUNIT)=ACTION
        GO TO 30
      ENDIF

C ... ORDERED TO TURN OFF BUT SITE IS ALREADY TURNED OFF.....

      IF (STATS(EWID(EWUNIT)) .EQ. 0) THEN
        IF (DEBUG .GE. 1) THEN
          WRITE(16,*) '   EW SITE #, STATUS',
     &      EWUNIT,STATS(EWID(EWUNIT))
          WRITE(16,*) '   ORDERED TO TURN OFF BUT ALREADY OFF'
        ENDIF
        STATS(EWID(EWUNIT))=ACTION
        EWSTATS(EWUNIT)=ACTION
        GO TO 30
      ENDIF

C ... BEFORE SHUTTING OFF THE EW SITE .....
C ... SET ALL DETECTED TARGETS TO UNDETECTED STATUS .....

      DO 110 I=1,NAC
      UPEWTGT(EWUNIT,I)=-EWDETECT(EWUNIT,I)
      EWDETECT(EWUNIT,I)=0
110    CONTINUE

      J1=EWID(EWUNIT)

      IF (DEBUG .GE. 3) THEN
        WRITE(16,*) '   EW SITE INPUT DELAY PARAMETERS:'
        WRITE(16,*) EWPROCDLY(EWUNIT),EWDAMDLY(EWUNIT),
     &    EWDAMRAT(EWUNIT),PS(J1)
      ENDIF

      CALL CDELAY(EWPROCDLY(EWUNIT),EWDAMDLY(EWUNIT),
     &  EWDAMRAT(EWUNIT),PS(J1),DELAY)

      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) '   CHANGE IN DETECTION STATUS FOR EW SITE #',
     &    EWUNIT
        WRITE(16,*) '   (-1= LOST, 0= NO CHANGE, 1= NEW DETECTION)'
        WRITE(16,*) '   (1-NAC) UPDATE DETECTION ARRAY :',
```

```
     &      (UPEWTGT(EWUNIT,L),L=1,NAC)
            WRITE(16,*) '   COMPUTED DELAY=',DELAY,
     &      '  TIME WHEN MESSAGE WILL BE SENT=',TIME+DELAY
          ENDIF
          IF (DEBUG .GE. 1) THEN
            WRITE(16,1003) TIME,EWUNIT,TIME+DELAY
 1003       FORMAT(F5.0,' SEC EW SITE ',I2,' WILL SEND MESSAGE',
     &      ' AT TIME ',F5.0)
          ENDIF

          CALL EWMES(EWUNIT,(TIME+DELAY))

C ... SET EW SITE STATUS TO OFF .....

          STATS(EWID(EWUNIT))=ACTION
          EWSTATS(EWUNIT)=ACTION

          IF(DEBUG .GE. 1) THEN
            WRITE(16,1001) CTIME,EWUNIT
 1001   FORMAT(F5.0,' SEC EW SITE ',I2,' ORDERED TO TURN OFF; DOES SO.')
          ENDIF

        ELSEIF(ACTION .EQ. 1) THEN

C ... ORDERED TO TURN ON BUT SITE IS ALREADY TURNED ON.....

          IF (STATS(EWID(EWUNIT)) .EQ. 1) THEN
            IF (DEBUG .GE. 1) THEN
              WRITE(16,*) '   EW SITE #, STATUS',
     &          EWUNIT,STATS(EWID(EWUNIT))
              WRITE(16,*) '   ORDERED TO TURN ON BUT ALREADY ON'
            ENDIF
            EWSTATS(EWUNIT)=ACTION
            GO TO 30
          ENDIF

          STATS(EWID(EWUNIT))=ACTION
          EWSTATS(EWUNIT)=ACTION

          IF(DEBUG .GE. 1) THEN
            WRITE(16,1002) CTIME,EWUNIT
 1002   FORMAT(F5.0,' SEC EW SITE ',I2,' ORDERED TO TURN ON; DOES SO.')
          ENDIF
        ENDIF

C ... DELETE RECEIVED MESSAGE FROM HOLDING ARRAY AND SHIFT UP OTHERS

 30   CALL SHIFTU(II)

        RETURN
```

```
      END
--------------------------------
      SUBROUTINE REWSTA(II)

C.....LATEST CHANGE MADE 07/25/91 BFS.....
C     FOR MESSAGES GOING TO EW REPORTING STATIONS ONLY, THIS SUB.
C     RECEIVES MESSAGE OFF HOLDING MESSAGE ARRAY AND PROCESSES IT.
C     EW REPORTING STATIONS ARE MESSAGES PASSER ONLY AND DO NO ACTION
C     OF THEMSELVES. THE SOURCE AND DESTINATION OF THE MESSAGE WILL
C     DETERMINE WHERE COPIES OF THE MESSAGE GET SENT. IF THE MESSAGES
C     ID # IS UNIQUE THE MESSAGE IS PASSED ALONG. IF IT IS NOT UNIQUE,
C     THE MESSAGE IS NOT PASSED ALONG, THUS AVOIDING INFINITE LOOPS AND
C     MULTIPLE MESSAGES.
C
C     CALLED BY - MSGRDY
C     CALLS     - NSPS,SHIFTU,CDELAY
C.....

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

      INTEGER II

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG
      REAL TIME,PST
C.....COMMON BLOCK.....

      INCLUDE 'QIADS DEF'
      INCLUDE 'EW1 DEF'
      INCLUDE 'HOLDM1 DEF'
      INCLUDE 'MESID DEF'
      INCLUDE 'CON1 DEF'
      INCLUDE 'RPTMES1 DEF'
      INCLUDE 'RPTDATA DEF'
      INCLUDE 'ADWDATA DEF'

      COMMON/DEBUG/DEBUG
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)

C.....DECLARE LOCAL VARIABLES.....
```

```
      INTEGER ACTION,JJ,L,LL,MESID,ORIGTYPE,POS,PREVTYPE,UNIT,TYPE,
     & ORIGUNIT,DESTTYPE,DESTUNIT
      REAL MSG(MAXMSGSIZE),DELAY,PSURVIV
C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                                  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C.....LOCAL TO THIS SUBROUTINE.....
C     ACTION - INTEGER INDICATING ACTION DESIRED IN MESSAGE
C     DELAY - TIME DELAY FROM PRESENT TIME WHEN MESSAGE WILL BE SENT
C     DESTTYPE - INTEGER HOLDING DESTINATION SOURCE MESSAGE IADS TYPE #
C     DESTUNIT - INTEGER HOLDING DESTINATION SOURCE MESSAGE IADS UNIT #
C     II - PASSED HOLDING MESSAGE ARRAY POINTER
C     L - LOOP COUNTER
C     LL - LOOP COUNTER
C     MSG(MAXMSGSIZE) - HOLDS RECEIVED MESSAGE INFORMATION
C     MESID - INCOMING MESSAGE ID #
C     ORIGTYPE - INTEGER HOLDING THE ORIGIN SOURCE MESSAGE IADS TYPE #
C     ORIGUNIT - INTEGER HOLDING THE ORIGIN SOURCE MESSAGE IADS UNIT #
C     POS - POSITION OF NEXT OPEN SLOT IN MESSAGE ID HOLDING ARRAY
C     PREVTYPE - INTEGER HOLDING THE PREVIOUS SOURCE IADS TYPE #
C     PSURVIV - CURRENT SURVIVAL PROBABILITY OF THIS UNIT #
C     TYPE - IADS EW REPORTING TYPE NUMBER (=2)
C     UNIT - HOLDS UNIT #
C.....START PROGRAM.....

C     WRITE(*,*) 'REWSTA'

C ... RECEIVE DATA FROM MESSAGE .....

      TYPE=2
      UNIT=NINT(MESHOLD(II,7))
      ACTION=NINT(MESHOLD(II,8))
      PREVTYPE=NINT(MESHOLD(II,4))
      MESID=NINT(MESHOLD(II,9))
      ORIGTYPE=NINT(MESHOLD(II,10))
      ORIGUNIT=NINT(MESHOLD(II,11))
      DESTTYPE=NINT(MESHOLD(II,12))
      DESTUNIT=NINT(MESHOLD(II,13))

C ... DETERMINE IF THIS STATION IS ALIVE .....

      CALL NSPS(TYPE,UNIT,PSURVIV)
      IF(PSURVIV .LT. MINRPTPS) THEN
        IF(DEBUG .GE. 1) THEN
         WRITE(16,*)'   RPT STA',UNIT,'DEAD; DOES NOT RECEIVE ANY',
```

```
     &    ' MESSAGES OR SEND ANY'
          WRITE(16,*)'   PSURVIVAL=',PSURVIV,' MIN PS=',MINRPTPS
        ENDIF
        RETURN
      ENDIF

C ... CONTINUE IF STATION IS NOT DEAD.....

      IF(DEBUG .GE. 2) THEN
        WRITE(16,*)'   IN REWSTA, RPT STA',UNIT,'RECEIVES ',
     &  'MES ID#=',MESID
      ENDIF

C ... COMPARE INCOMING MESSAGE ID WITH ID # OF PREVIOUSLY RECEIVED
C ... MESSAGES, IF A MATCH; IGNORE MESSAGE, TAKE IT OFF HOLDING ARRAY

      POS=RPTMESIDPOS(UNIT)
      DO 10 JJ=1,POS-1
      IF(RPTMESID(UNIT,JJ) .EQ. MESID) THEN
        IF(DEBUG .GE. 2) THEN
          WRITE(16,*)'   MESSAGE ID# ',MESID,' IGNORED;',
     &       ' MATCHES WITH PREVIOUS ID#'
        ENDIF
        CALL SHIFTU(II)
        RETURN
      ENDIF
   10 CONTINUE

C ... TO GET HERE, THIS IS NEW MESSAGE SO HOLD ITS ID # .....

C ... CHECK THAT YOU HAVE ROOM TO ADD ANOTHER ID # TO ID ARRAY.....

      IF(POS .GT. MAXRPTMESID) THEN
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(*,*)  'RPT STA #',UNIT,' NEEDS TO STORE NEW ID # AND',
     &  ' STA ID ARRAY IS FULL'
        WRITE(16,*) 'RPT STA #',UNIT,' NEEDS TO STORE NEW ID # AND',
     &  ' STA ID ARRAY IS FULL'
        WRITE(*,*)  'PRESENT VALUE OF MAXRPTMESID PARAMETER IS',POS-1
        WRITE(16,*) 'PRESENT VALUE OF MAXRPTMESID PARAMETER IS',POS-1
        WRITE(*,*)  'INCREASE MAXRPTMESID IN RPTMES1 DEF AND',
     &  ' RECOMPILE PROGRAM MODULES'
        WRITE(16,*) 'INCREASE MAXRPTMESID IN RPTMES1 DEF AND',
     &  ' RECOMPILE PROGRAM MODULES'
        WRITE(16,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
      STOP
      ENDIF

C ... ARRAY HAS ROOM SO BUT NEW MSG ID # INTO IT AND INCREMENT POINTER
```

```
      RPTMESID(UNIT,POS)=MESID
      RPTMESIDPOS(UNIT)=RPTMESIDPOS(UNIT)+1

C.....TRANSFER DATA INTO LOCAL HOLDING ARRAY.....

      DO 15 JJ=1,MAXMSGSIZE
      MSG(JJ)=MESHOLD(II,JJ)
   15 CONTINUE

C.....DELETE CURRENT FROM HOLDING MESSAGE ARRAY ..... .

      CALL SHIFTU(II)

C ...DETERMINE THE ARRAY INDEX FOR THIS IADS ELEMENT IN THE NON-SAM
C ... TARGET PS ARRAY .....

      DO 17 JJ=1,NUMIADSTGTS
      IF(IDIADSTGT(JJ,2) .EQ. 2 .AND. IDIADSTGT(JJ,3) .EQ. UNIT) THEN
        L=IDIADSTGT(JJ,1)
        IF (DEBUG .GE. 2) THEN
          WRITE(16,*) '   EW REPORTING STATION #',UNIT,
     &     ' IS NON-SAM TARGET # =',L
        ENDIF
        GOTO 18
      ENDIF
   17 CONTINUE
   18 CONTINUE

C ... COMPUTE TIME DELAY OF MESSAGE .....

      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) '   INPUTED EW RPT STATION DELAY PARAMETERS:'
        WRITE(16,*) RPTPROCDLY(UNIT),RPTDAMDLY(UNIT),
     &   RPTDAMRAT(UNIT),PST(L)
      ENDIF
      CALL CDELAY(RPTPROCDLY(UNIT),RPTDAMDLY(UNIT),RPTDAMRAT(UNIT),
     &   PST(L),DELAY)
      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) '   COMPUTED RPT STATION DELAY=',DELAY,
     &      ' TIME WHEN MESSAGE WILL BE SENT=',TIME+DELAY
      ENDIF
      IF (DEBUG .GE. 1) THEN
        WRITE(16,1003) TIME,UNIT,TIME+DELAY,MESID
 1003   FORMAT(F5.0,' SEC RPT STA ',I2,' WILL SEND MESSAGE',
     &      ' AT TIME ',F5.0,' MSGID # ',I2)
      ENDIF
      MSG(2)=TIME+DELAY

C.....DETERMINE SOURCE OF MESSAGE AND APPROPRIATE SENDING DESTINATION .
```

```
      IF(PREVTYPE .EQ. 1) THEN
C ... MESSAGE IS FROM AN EW SITE, SEND COPY OF MESSAGE TO REPORTING
C ... STATIONS AND ADWOC, IF ADWOC IS DEAD, SEND ALERT TO BAT AND SAMS.

C.....DETERMINE WHERE MESSAGE IS GOING VIA CONNECT ARRAY .....
C.... LOOK FIRST FOR A MATCH OF REPORT STATION TYPE AND UNIT # ..

      DO 20 L=1,NUMCON
      IF(CONNECT(L,1) .EQ. 2  .AND. CONNECT(L,2) .EQ. UNIT) THEN
        IF(CONNECT(L,3) .EQ. 2) THEN
          MSG(4)=REAL(CONNECT(L,1))
          MSG(5)=REAL(CONNECT(L,2))
          MSG(6)=REAL(CONNECT(L,3))
          MSG(7)=REAL(CONNECT(L,4))
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND MES.',
     &        ' TO RPT STA #',NINT(MSG(7)),'MSGID=',NINT(MSG(9))
          ENDIF
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &        (MSG(LL),LL=1,MAXMSGSIZE)
          ENDIF
          CALL HOLDMES(MSG)
        ENDIF
        IF(CONNECT(L,3) .EQ. 3) THEN
          MSG(4)=REAL(CONNECT(L,1))
          MSG(5)=REAL(CONNECT(L,2))
          MSG(6)=REAL(CONNECT(L,3))
          MSG(7)=REAL(CONNECT(L,4))
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND MES.',
     &      ' TO ADWOC; MSGID=',NINT(MSG(9))
          ENDIF
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &        (MSG(LL),LL=1,MAXMSGSIZE)
          ENDIF
          CALL HOLDMES(MSG)
        ENDIF
C ... IF ADWOC IS DEAD THEN SEND AN ALERT TO THE BATTALION CENTERS
        IF(PSADWOC .LT. MINADWOCPS) THEN
          MSG(9)=MSGID
          IF(CONNECT(L,3) .EQ. 4) THEN
            MSG(4)=REAL(CONNECT(L,1))
            MSG(5)=REAL(CONNECT(L,2))
            MSG(6)=REAL(CONNECT(L,3))
            MSG(7)=REAL(CONNECT(L,4))
C ... SET ALERT MESSAGE BIT.....
            MSG(8)=1.0
C ... SET NEW ORGIN AND DESTINATION IADS TYPE AND UNIT #S.....
```

```
            MSG(10)=REAL(CONNECT(L,1))
            MSG(11)=REAL(CONNECT(L,2))
            MSG(12)=REAL(CONNECT(L,3))
            MSG(13)=REAL(CONNECT(L,4))
            IF (DEBUG .GE. 2) THEN
            WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND MES.',
     &  ' TO BAT. #',NINT(MSG(7)),'MSGID=',NINT(MSG(9)),' ** ALERT **'
            ENDIF
            IF (DEBUG .GE. 2) THEN
              WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &          (MSG(LL),LL=1,MAXMSGSIZE)
            ENDIF
            CALL HOLDMES(MSG)
C ..UNSET ALERT MESSAGE BIT AND ORIGIN/DESTINATION .....
            MSG(8)=ACTION
            MSG(10)=ORIGTYPE
            MSG(11)=ORIGUNIT
            MSG(12)=DESTTYPE
            MSG(13)=DESTUNIT
          ENDIF
          IF(CONNECT(L,3) .EQ. 5) THEN
            MSG(4)=REAL(CONNECT(L,1))
            MSG(5)=REAL(CONNECT(L,2))
            MSG(6)=REAL(CONNECT(L,3))
            MSG(7)=REAL(CONNECT(L,4))
C ... SET ALERT MESSAGE BIT.....
            MSG(8)=1.0
C ... SET NEW ORGIN AND DESTINATION IADS TYPE AND UNIT #S.....
            MSG(10)=REAL(CONNECT(L,1))
            MSG(11)=REAL(CONNECT(L,2))
            MSG(12)=REAL(CONNECT(L,3))
            MSG(13)=REAL(CONNECT(L,4))
            IF (DEBUG .GE. 2) THEN
          WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND MES. TO',
     & ' SAM SITE #',NINT(MSG(7)),'MSGID=',NINT(MSG(9)),' ** ALERT **'
            ENDIF
            IF (DEBUG .GE. 2) THEN
              WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &          (MSG(LL),LL=1,MAXMSGSIZE)
            ENDIF
            CALL HOLDMES(MSG)
C ..UNSET ALERT MESSAGE BIT AND ORIGIN/DESTINATION .....
            MSG(8)=ACTION
            MSG(10)=ORIGTYPE
            MSG(11)=ORIGUNIT
            MSG(12)=DESTTYPE
            MSG(13)=DESTUNIT
          ENDIF
          MSG(9)=MESID
        ENDIF
```

```
      ENDIF
   20 CONTINUE

      ELSEIF(PREVTYPE .EQ. 2) THEN

C ... MESSAGE COMES FROM AN EW REPORTING STATION. CHECK DESTINATION,
C ... SEND COPY OF MESSAGE BASED ON ULTIMATE ORIGIN AND DESTINATION

      IF(ORIGTYPE .EQ. 1) THEN

C.... MESSAGE IS FROM AN EW SITE. PASS MESSAGE ALONG TO ADWOC
C ... AND IF ADWOC IS DEAD, SEND AN ALERT TO BATTALION STATIONS
C.....DETERMINE WHERE MESSAGE IS GOING VIA CONNECT ARRAY .....
C.... LOOK FIRST FOR A MATCH OF REPORT STATION TYPE AND UNIT # ..

      DO 30 L=1,NUMCON
      IF(CONNECT(L,1) .EQ. 2  .AND. CONNECT(L,2) .EQ. UNIT) THEN
        IF(CONNECT(L,3) .EQ. 2) THEN
          MSG(4)=REAL(CONNECT(L,1))
          MSG(5)=REAL(CONNECT(L,2))
          MSG(6)=REAL(CONNECT(L,3))
          MSG(7)=REAL(CONNECT(L,4))
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND MES.',
     &        ' TO RPT STA #',NINT(MSG(7)),'MSGID',NINT(MSG(9))
          ENDIF
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &        (MSG(LL),LL=1,MAXMSGSIZE)
          ENDIF
          CALL HOLDMES(MSG)
        ENDIF
        IF(CONNECT(L,3) .EQ. 3) THEN
          MSG(4)=REAL(CONNECT(L,1))
          MSG(5)=REAL(CONNECT(L,2))
          MSG(6)=REAL(CONNECT(L,3))
          MSG(7)=REAL(CONNECT(L,4))
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND',
     &      ' MESSAGE TO ADWOC; MSGID=',NINT(MSG(9))
          ENDIF
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &        (MSG(LL),LL=1,MAXMSGSIZE)
          ENDIF
          CALL HOLDMES(MSG)
        ENDIF
C ... IF ADWOC IS DEAD THEN SEND AN ALERT TO THE BATTALION CENTERS
        IF(PSADWOC .LT. MINADWOCPS) THEN
          MSG(9)=MSGID
```

```
            IF(CONNECT(L,3) .EQ. 4) THEN
              MSG(4)=REAL(CONNECT(L,1))
              MSG(5)=REAL(CONNECT(L,2))
              MSG(6)=REAL(CONNECT(L,3))
              MSG(7)=REAL(CONNECT(L,4))
C ... SET ALERT MESSAGE BIT.....
              MSG(8)=1.0
C ... SET NEW ORGIN AND DESTINATION IADS TYPE AND UNIT #S.....
              MSG(10)=REAL(CONNECT(L,1))
              MSG(11)=REAL(CONNECT(L,2))
              MSG(12)=REAL(CONNECT(L,3))
              MSG(13)=REAL(CONNECT(L,4))
              IF (DEBUG .GE. 2) THEN
            WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND MES.',
     &  ' TO BAT. #',NINT(MSG(7)),'MSGID=',NINT(MSG(9)),' ** ALERT **'
              ENDIF
              IF (DEBUG .GE. 2) THEN
                WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &            (MSG(LL),LL=1,MAXMSGSIZE)
              ENDIF
              CALL HOLDMES(MSG)
C ..UNSET ALERT MESSAGE BIT AND ORIGIN/DESTINATION .....
              MSG(8)=ACTION
              MSG(10)=ORIGTYPE
              MSG(11)=ORIGUNIT
              MSG(12)=DESTTYPE
              MSG(13)=DESTUNIT
            ENDIF
            IF(CONNECT(L,3) .EQ. 5) THEN
              MSG(4)=REAL(CONNECT(L,1))
              MSG(5)=REAL(CONNECT(L,2))
              MSG(6)=REAL(CONNECT(L,3))
              MSG(7)=REAL(CONNECT(L,4))
C ... SET ALERT MESSAGE BIT.....
              MSG(8)=1.0
C ... SET NEW ORGIN AND DESTINATION IADS TYPE AND UNIT #S.....
              MSG(10)=REAL(CONNECT(L,1))
              MSG(11)=REAL(CONNECT(L,2))
              MSG(12)=REAL(CONNECT(L,3))
              MSG(13)=REAL(CONNECT(L,4))
              IF (DEBUG .GE. 2) THEN
            WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND MES. TO',
     & ' SAM SITE #',NINT(MSG(7)),'MSGID=',NINT(MSG(9)),' ** ALERT **'
              ENDIF
              IF (DEBUG .GE. 2) THEN
                WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &            (MSG(LL),LL=1,MAXMSGSIZE)
              ENDIF
              CALL HOLDMES(MSG)
C ..UNSET ALERT MESSAGE BIT AND ORIGIN/DESTINATION .....
```

```
            MSG(8)=ACTION
            MSG(10)=ORIGTYPE
            MSG(11)=ORIGUNIT
            MSG(12)=DESTTYPE
            MSG(13)=DESTUNIT
          ENDIF
          MSG(9)=MESID
C          MSGID=MSGID+1
C          IF (DEBUG .GE. 2) THEN
C            WRITE(16,*) ' MSGID ID # INCREMENTED TO =',MSGID,
C     &          ' FOR NEXT MESSAGE'
C          ENDIF
        ENDIF
      ENDIF
   30 CONTINUE

      ELSEIF(ORIGTYPE .EQ. 3) THEN

C.... MESSAGE IS FROM ADWOC CENTER. PASS MESSAGE ALONG TO EW SITE
C ... DESIGNATED AND TO OTHER EW REPORTING STATIONS

C ... SEND MESSAGE TO THE DESIGNATED EW SITE ONLY .....
      MSG(4)=REAL(CONNECT(L,1))
      MSG(5)=REAL(CONNECT(L,2))
      MSG(6)=MSG(12)
      MSG(7)=MSG(13)
      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND MESSAGE TO',
     &   ' EW SITE #',NINT(MSG(13)),'MSGID=',NINT(MSG(9))
      ENDIF
      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &   (MSG(LL),LL=1,MAXMSGSIZE)
      ENDIF
      CALL HOLDMES(MSG)

C... PASS ONLY MESSAGE TO OTHER EW REPORTING STATIONS

      DO 40 L=1,NUMCON
      IF(CONNECT(L,1) .EQ. 2  .AND. CONNECT(L,2) .EQ. UNIT) THEN
        IF(CONNECT(L,3) .EQ. 2) THEN
          MSG(4)=REAL(CONNECT(L,1))
          MSG(5)=REAL(CONNECT(L,2))
          MSG(6)=REAL(CONNECT(L,3))
          MSG(7)=REAL(CONNECT(L,4))
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND MES.',
     &       ' TO RPT STA #',NINT(MSG(7)),'MSGID=',NINT(MSG(9))
          ENDIF
          IF (DEBUG .GE. 2) THEN
```

```
            WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &        (MSG(LL),LL=1,MAXMSGSIZE)
          ENDIF
          CALL HOLDMES(MSG)
        ENDIF
      ENDIF
   40 CONTINUE
      ENDIF

      ELSEIF(PREVTYPE .EQ. 3) THEN
C ... SOURCE OF MESSAGE IS AN ADWOC CENTER

C.... MESSAGE IS FROM ADWOC CENTER. PASS MESSAGE ALONG TO EW SITE
C ... DESIGNATED AND TO OTHER EW REPORTING STATIONS

C ... SEND MESSAGE TO THE DESIGNATED EW SITE ONLY .....
      MSG(4)=REAL(CONNECT(L,1))
      MSG(5)=REAL(CONNECT(L,2))
      MSG(6)=MSG(12)
      MSG(7)=MSG(13)
      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND MESSAGE',
     &   ' TO EW SITE #',NINT(MSG(7)),'MSGID=',NINT(MSG(9))
      ENDIF
      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &   (MSG(LL),LL=1,MAXMSGSIZE)
      ENDIF
      CALL HOLDMES(MSG)

C... PASS ONLY MESSAGE TO OTHER EW REPORTING STATIONS

      DO 50 L=1,NUMCON
      IF(CONNECT(L,1) .EQ. 2  .AND. CONNECT(L,2) .EQ. UNIT) THEN
        IF(CONNECT(L,3) .EQ. 2) THEN
          MSG(4)=REAL(CONNECT(L,1))
          MSG(5)=REAL(CONNECT(L,2))
          MSG(6)=REAL(CONNECT(L,3))
          MSG(7)=REAL(CONNECT(L,4))
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) 'RPT STA #',NINT(MSG(5)),' WILL SEND MES.',
     &       ' TO RPT STA #',NINT(MSG(7)),'MSGID=',NINT(MSG(9))
          ENDIF
          IF (DEBUG .GE. 2) THEN
            WRITE(16,*) '   MESSAGE TO BE SEND IS:',
     &       (MSG(LL),LL=1,MAXMSGSIZE)
          ENDIF
          CALL HOLDMES(MSG)
        ENDIF
      ENDIF
```

```
 50 CONTINUE

      ELSEIF(PREVTYPE .EQ. 4) THEN

C ... SOURCE OF MESSAGE IS A BATTALION CENTER
C ... PRESENTLY NO CONNECTION ASSUMED

      ELSEIF(PREVTYPE .EQ. 5) THEN

C ... SOURCE OF MESSAGE IS A SAM UNIT
C ... PRESENTLY NO CONNECTION ASSUMED

      ENDIF

      RETURN
      END
---------------------------------
      SUBROUTINE RSAM(II)

C.....LATEST CHANGE MADE 07/24/91 BFS.....
C     FOR MESSAGES GOING TO SAM UNITS. RECEIVES
C     A MESSAGE OFF HOLDING MESSAGE ARRAY AND PROCESSES IT.
C     IF THE INCOMING MESSAGES ID # IS UNIQUE, THE
C     MESSAGE AND INFORMATION IS RECEIVED. IF THE ID # IS NOT UNIQUE,
C     THE MESSAGE IS REJECTED, THUS AVOIDING REPETITIVE ACTIONS.
C
C     CALLED BY - MSGRDY
C     CALLS     - SHIFTU,HOLDMES
C.....


      PARAMETER (IM=30,JM=25,KM=20)


      IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

      INTEGER II

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG,STATS
      REAL PS,PSSMIN

C.....COMMON BLOCK.....

      INCLUDE 'HOLDM1 DEF'
      INCLUDE 'SAMMES1 DEF'
      INCLUDE 'SAMDATA DEF'
```

```
      COMMON/DEBUG/DEBUG
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER ACTION,JJ,MESID,POS,SAMNUM
      REAL MSG(MAXMSGSIZE),CTIME,DELAY

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     PS(JM) - SURVIVAL PROBABILITY OF SITE JM
C     PSSMIN - MIN. SURVIVAL PROBABILITY FOR SAM SITE TO STAY IN SIM.
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C               - IF =0; SITE IS NOT RADIATING
C               - IF =1; SITE IS RADIATING
C.....LOCAL TO THIS SUBROUTINE.....
C     ACTION - INTEGER SPECIFYING ACTION DESIRED FROM THIS MESSAGE
C     CTIME - CURRENT TIME: TIME THAT MESSAGE IS TO BE SENT
C     II - PASSED HOLDING MESSAGE ARRAY POINTER
C     JJ - LOOP COUNTER
C     MSG(MAXMSGSIZE) - HOLDS RECEIVED MESSAGE INFORMATION
C     MESID - INCOMING MESSAGE ID #
C     ORIGTYPE - INTEGER HOLDING THE ORIGIN SOURCE MESSAGE IADS TYPE #
C     ORIGUNIT - INTEGER HOLDING THE ORIGIN SOURCE MESSAGE IADS UNIT #
C     POS - POSITION OF NEXT OPEN SLOT IN MESSAGE ID HOLDING ARRAY
C     PREVTYPE - INTEGER HOLDING THE PREVIOUS SOURCE IADS TYPE #
C     SAMNUM - INTEGER HOLDING THE SAM UNIT # THIS MESSAGE IS FOR
C.....START PROGRAM.....

C     WRITE(*,*) 'RSAM'

C ... RECEIVE DATA FROM MESSAGE .....

      CTIME=MESHOLD(II,2)
      SAMNUM=NINT(MESHOLD(II,7))
      ACTION=NINT(MESHOLD(II,8))
      MESID=NINT(MESHOLD(II,9))
```

```
C ... DETERMINE IF THIS SAM SITE IS ALIVE .....

      IF (PS(SAMNUM) .LT. PSSMIN) THEN
        IF (DEBUG .GE. 1) THEN
         WRITE(16,*)'   SAM SITE',SAMNUM,'DEAD; DOES NOT RECEIVE ANY',
     &    ' MESSAGES OR SEND ANY'
         WRITE(16,*)'   PSURVIVAL=',PS(SAMNUM),' MIN PS=',PSSMIN
        ENDIF
        RETURN
      ENDIF

C ... CONTINUE IF SAM SITE IS NOT DEAD.....

      IF(DEBUG .GE. 2) THEN
        WRITE(16,*)'   IN RSAM, RECEIVING MES ID#=',MESID
      ENDIF

C ... COMPARE INCOMING MESSAGE ID WITH ID # OF PREVIOUSLY RECEIVED
C ... MESSAGES, IF A MATCH; IGNORE MESSAGE, TAKE IT OFF HOLDING ARRAY

      POS=RSAMMESIDPOS(SAMNUM)

      DO 10 JJ=1,POS-1
      IF(RSAMMESID(SAMNUM,JJ) .EQ. MESID) THEN
        IF(DEBUG .GE. 2) THEN
          WRITE(16,*)'   MESSAGE #',MESID,' IGNORED;',
     &      ' MATCHES WITH PREVIOUS ID#'
        ENDIF
        CALL SHIFTU(II)
        RETURN
      ENDIF
   10 CONTINUE

C ... TO GET HERE, THIS IS NEW MESSAGE SO HOLD ITS ID # .....

C ... CHECK THAT YOU HAVE ROOM TO ADD ANOTHER ID # TO ID ARRAY.....

      IF(POS .GT. MAXSAMMESID) THEN
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(*,*) 'SAM SITE #',SAMNUM,' NEEDS TO STORE',
     &   ' NEW ID # AND SAM SITE ID ARRAY IS FULL'
        WRITE(*,*) 'PRESENT VALUE OF MAXSAMMESID PARAMETER IS',POS-1
        WRITE(*,*) 'INCREASE MAXADWMESID IN SAMMES1 DEF AND',
     &   ' RECOMPILE PROGRAM MODULES'
        WRITE(*,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(16,*) ' ***    PROGRAM STOPPED    ***'
        WRITE(16,*) 'SAM SITE #',SAMNUM,' NEEDS TO STORE',
     &   ' NEW ID # AND SAM SITE ID ARRAY IS FULL'
        WRITE(16,*) 'PRESENT VALUE OF MAXSAMMESID PARAMETER IS',POS-1
        WRITE(16,*) 'INCREASE MAXADWMESID IN SAMMES1 DEF AND',
```

```
&   ' RECOMPILE PROGRAM MODULES'
    WRITE(16,*) ' ***    PROGRAM STOPPED    ***'
    STOP
    ENDIF

C ... ARRAY HAS ROOM SO PUT NEW MSG ID # INTO IT AND INCREMENT POINTER

    RSAMMESID(SAMNUM,POS)=MESID
    RSAMMESIDPOS(SAMNUM)=RSAMMESIDPOS(SAMNUM)+1
    WRITE(16,*) '   MESID#',MESID,'STORED AT SAM SITE #',SAMNUM,
   & 'NEW MES POS = ',RSAMMESIDPOS(SAMNUM)


    IF(ACTION .EQ. 0) THEN

C ... ORDERED TO TURN OFF BUT IS ALREADY OFF BECAUSE IT IS DEAD.....

      IF (PS(SAMNUM) .LT. PSSMIN) THEN
        IF (DEBUG .GE. 1) THEN
          WRITE(16,*) '  SAM SITE #,PS,PSSMIN',
   &        SAMNUM,PS(SAMNUM),PSSMIN
          WRITE(16,*) '   ORDERED TO TURN OFF BUT ALREADY DEAD'
        ENDIF
        STATS(SAMNUM)=ACTION
        GO TO 30
      ENDIF

C ... ORDERED TO TURN OFF BUT SITE IS ALREADY TURNED OFF.....

      IF (STATS(SAMNUM) .EQ. 0) THEN
        IF (DEBUG .GE. 1) THEN
          WRITE(16,*) '   SAM SITE #, STATUS',
   &        SAMNUM,STATS(SAMNUM)
          WRITE(16,*) '   ORDERED TO TURN OFF BUT ALREADY OFF'
        ENDIF
        GO TO 30
      ENDIF

      IF (DEBUG .GE. 3) THEN
        WRITE(16,*) '   SAM SITE INPUT DELAY PARAMETERS:'
        WRITE(16,*) SAMPROCDLY,SAMDAMDLY,
   &     SAMDAMRAT,PS(SAMNUM)
      ENDIF

      CALL CDELAY(SAMPROCDLY,SAMDAMDLY,SAMDAMRAT,PS(SAMNUM),DELAY)

      IF (DEBUG .GE. 2) THEN
        WRITE(16,*) '   COMPUTED DELAY=',DELAY,
   &     '  TIME WHEN MESSAGE WILL BE SENT=',CTIME+DELAY
      ENDIF
```

```
      IF (DEBUG .GE. 1) THEN
        WRITE(16,1003) CTIME,SAMNUM,CTIME+DELAY
1003     FORMAT(F5.0,' SEC SAM SITE ',I2,' WILL SEND MESSAGE',
     &    ' AT TIME ',F5.0)
      ENDIF

C ... CONSTRUCT A MESSAGE WHEN SITE MUST SHUT DOWN AND SEND IT TO ?

C      CALL HOLDMES(MSG)

C ... SET SAM SITE STATUS TO OFF .....

      STATS(SAMNUM)=ACTION

      IF(DEBUG .GE. 1) THEN
        WRITE(16,1001) CTIME,SAMNUM
1001   FORMAT(F5.0,' SEC SAM SITE ',I2,' ORDERED TO TURN OFF; DOES.')
      ENDIF

    ELSEIF(ACTION .EQ. 1) THEN

C ... ORDERED TO TURN ON BUT SITE IS ALREADY TURNED ON.....

      IF (STATS(SAMNUM) .EQ. 1) THEN
        IF (DEBUG .GE. 1) THEN
          WRITE(16,*) '   SAM SITE #, STATUS',
     &      SAMNUM,STATS(SAMNUM)
          WRITE(16,*) '   ORDERED TO TURN ON BUT ALREADY ON'
        ENDIF
        GO TO 30
      ENDIF

      STATS(SAMNUM)=ACTION

      IF(DEBUG .GE. 1) THEN
        WRITE(16,1002) CTIME,SAMNUM
1002   FORMAT(F5.0,' SEC SAM SITE ',I2,' ORDERED TO TURN ON; DOES.')
      ENDIF
    ENDIF

C.....DELETE CURRENT MESSAGE FROM HOLDING MESSAGE ARRAY .....

 30  CONTINUE

    CALL SHIFTU(II)


    RETURN
    END
-----------------------------------
```

```
      SUBROUTINE SETID

C.....LATEST CHANGE MADE 7/09/91 BFS.....
C     SETS THE ARRAY INDICATING WHETHER A NON-SAM TARGET IS AN IADS
C     ELEMENT(=1) OR NOT(=0)
C
C     CALLED BY - REDIAD
C     CALLS     - NONE
C.....

      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER JJ,KK

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER NTARGETS

C.....COMMON BLOCK.....

      INCLUDE 'CON1 DEF'

      COMMON/WEAPONS/BOM(IM),BOMPT(IM,4),BOMTGT(IM,2,4),BOMTIM(IM,4),
     $BOMPK(2,20),NTARGETS,PSAL(IM,2,4),LNCH1(IM),LNCH2(IM),
     $L1CKPT(IM,4),L2CKPT(IM,4),L1WPN(IM,4),L2WPN(IM,4),LNCH1TIM(IM,4),
     $CNTRL2(IM),C2ETIM(IM,4),C2START(IM,4),C2END(IM,4),C2WPN(IM,4),
     $C2TIME(IM,4),LNCH2TIM(IM,4),C2STIM(IM,4),PSAC(IM,4),WHO(IM,2,2,4),
     $C2DEDTIM(IM)

C.....DESCRIPTION OF VARIABLES IN THIS SUBROUTINE.....
C.....                                  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....VARIABLES IN COMMON BLOCK USED IN THIS SUBROUTINE .....
C   NTARGETS - MAXIMUM NUMBER OF NON-SAM TARGETS IN THIS SIM
C.....VARIABLES LOCAL TO THIS SUBROUTINE.....
C     JJ - LOOP COUNTER
C     KK - LOOP COUNTER
C.....START PROGRAM.....

C     WRITE(*,*) 'SETID'

      DO 10 JJ=1,NTARGETS
      IADSTGT(JJ)=0
```

```
      DO 20 KK=1,NUMIADSTGTS
      IF(IDIADSTGT(KK,1) .EQ. JJ) THEN
        IADSTGT(JJ)=1
      ENDIF
   20 CONTINUE
   10 CONTINUE

C     WRITE(*,*) 'JJ,IADSTGT(JJ)'
C     DO 30 JJ=1,NTARGETS
C     WRITE(*,*) JJ,IADSTGT(JJ)
C  30 CONTINUE

      RETURN
      END
--------------------------------
      SUBROUTINE SHIFTU(I)

C.....LATEST CHANGE MADE 07/03/91 BFS.....
C     SHIFTS WAITING MESSAGES ONE POSITION UP
C
C     CALLED BY - MSGRDY
C     CALLS     - NONE
C.....


      PARAMETER (IM=30,JM=25,KM=20)


      IMPLICIT LOGICAL (A-Z)

C.....DECLARE PASSED VARIABLES.....

      INTEGER I

C.....DECLARE LOCAL VARIABLES.....

      INTEGER II,L,POS

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG

C.....COMMON BLOCK.....

      INCLUDE 'HOLDM1 DEF'
      COMMON/DEBUG/DEBUG

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....IN COMMON BLOCK.....
```

```
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C.....LOCAL TO THIS SUBROUTINE.....
C     II - LOOP COUNTER
C     L - LOOP COUNTER
C     POS - INPUT INDEX OF MESSAGE IN HOLDING ARRAY
C.....START PROGRAM.....

C     WRITE(*,*) 'SHIFTU'


C... SHIFT PENDING MESSAGES UP ONE POSITION, CLEAR DATA AT LAST
C ... POINTER POSITION AND DECREMENTS POINTER BY ONE .....

      POS=NEXTHOLDPOS

      DO 30 L=I,POS
      DO 40 II=1,MAXMSGSIZE
      MESHOLD(L,II)=MESHOLD(L+1,II)
   40 CONTINUE
   30 CONTINUE

      I=I-1
      NEXTHOLDPOS=NEXTHOLDPOS-1

      IF(DEBUG .GE. 3) THEN
        WRITE(16,*) 'HOLDING MESSAGES SHIFTED UP, NEXT MESSAGE',
     &   ' POINTER REDUCED TO =',NEXTHOLDPOS
      ENDIF

      RETURN
      END
---------------------------------
      SUBROUTINE EWOFF

C.....LATEST CHANGE MADE 6/28/91 BFS.....
C     TURNS OFF AN EW SITE AT A GIVEN TIME DURING THE SIMULATION.
C     USED TO TEST THE IADS MODIFICATIONS.
C
C     CALLED BY - IADS
C     CALLS     - HOLDMES
C.....

      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER II
      REAL MSG(13)
```

```
C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG,STATS
      REAL TIME
C.....COMMON BLOCK.....

      INCLUDE 'QIADS DEF'
      INCLUDE 'EW1 DEF'
      INCLUDE 'MESID DEF'

      COMMON/DEBUG/DEBUG
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....VARIABLES IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C               - IF =0; SITE IS NOT RADIATING
C               - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C               - AND GETTING READY TO FIRE, SITE IS RADIATING
C               - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C               - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C.....LOCAL TO THIS SUBROUTINE.....
C     MSG(13) - HOLDS DATA CONTAINING MESSAGE TO BE SENT
C     II
C.....START PROGRAM.....

C     WRITE(*,*) 'EWOFF'

      MSG(1)=TIME
      MSG(2)=TIME+30.0
      MSG(3)=1.0
      MSG(4)=3.0
      MSG(5)=1.0
      MSG(6)=1.0
      MSG(7)=1.0
      MSG(8)=0.0
      MSG(9)=REAL(MSGID)
      MSG(10)=3.0
      MSG(11)=1.0
      MSG(12)=1.0
      MSG(13)=1.0
```

```
      IF (DEBUG .GE. 1) THEN
        WRITE(16,1003) MSG(1),MSG(2),NINT(MSG(7))
 1003   FORMAT(F5.0,' SEC ADWOC WILL SEND SHUT OFF MESSAGE AT TIME',
     &  F5.0,' TO EW SITE # ',I2)
        WRITE(16,*) '         MSGID=',NINT(MSG(9))
      ENDIF
      CALL HOLDMES(MSG)
      MSGID=MSGID+1

      RETURN
      END
--------------------------------
      SUBROUTINE MESSUM

C.....LATEST CHANGE MADE 07/24/91 BFS.....
C     USED AS AN INFORMATION, STATUS AND DEBUG GUIDE.
C     LISTS THE FINAL MESSAGE POINTER POSITIONS AND MAX LIMITS
C     FOR THE IADS ELEMENTS.
C
C     CALLED BY - TACOPS
C     CALLS     - NONE
C.....


      PARAMETER (IM=30,JM=25,KM=20)


      IMPLICIT LOGICAL (A-Z)

C.....'....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG,NSITE

C.....COMMON BLOCK.....

C      INCLUDE 'HOLDM1 DEF'
       INCLUDE 'EWMES2 DEF'
       INCLUDE 'RPTMES1 DEF'
       INCLUDE 'ADWMES1 DEF'
C      INCLUDE 'BATMES1 DEF'
       INCLUDE 'QIADS  DEF'
       INCLUDE 'SAMMES1 DEF'
       INCLUDE 'RPTDATA DEF'

       COMMON/DEBUG/DEBUG
       COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
      $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)

C.....DECLARE LOCAL VARIABLES.....
```

```
      INTEGER II,JJ,POS

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                              .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PENETRATOR CHECKPOINTS
C.....IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C.....LOCAL TO THIS SUBROUTINE.....
C     II - LOOP COUNTER
C     JJ - LOOP COUNTER
C.....START PROGRAM.....

C     WRITE(*,*) 'MESSUM'

      WRITE(16,*) '------MESSAGE POINTER SUMMARY----------'

      WRITE(16,*) '------------ EW SITES----------'
      WRITE(16,*) 'MAX # OF ID #S POSSIBLE TO BE STORED =',MAXEWMESID
      DO 10 JJ=1,NEWSITES
      IF(REWMESIDPOS(JJ) .GE. 1) THEN
        IF(DEBUG .GE. 1) THEN
          WRITE(16,*)' EW SITE #',JJ,' FINAL MESSAGE POINTER=',
     &      REWMESIDPOS(JJ)
        ENDIF
      ENDIF
   10 CONTINUE

      WRITE(16,*) '--------EW REPORTING STATIONS----------'
      WRITE(16,*) 'MAX # OF ID #S POSSIBLE TO BE STORED =',MAXRPTMESID
      DO 20 JJ=1,NUMRPT
      IF(RPTMESIDPOS(JJ) .GE. 1) THEN
        IF(DEBUG .GE. 1) THEN
          WRITE(16,*)' EW RPT SITE #',JJ,' FINAL MESSAGE POINTER=',
     &      RPTMESIDPOS(JJ)
        ENDIF
      ENDIF
   20 CONTINUE

      WRITE(16,*) '-------- ADWOC CENTER ----------'
      WRITE(16,*) 'MAX # OF ID #S POSSIBLE TO BE STORED =',MAXADWMESID
      IF(ADWMESIDPOS .GE. 1) THEN
        IF(DEBUG .GE. 1) THEN
          WRITE(16,*)' ADWOC CENTER FINAL MESSAGE POINTER=',
     &      ADWMESIDPOS
        ENDIF
      ENDIF
```

```
      WRITE(16,*) '------------SAM SITES--------------------------'
      WRITE(16,*) 'MAX # OF ID #S POSSIBLE TO BE STORED =',MAXSAMMESID
      DO 100 JJ=NEWSITES+1,NSITE
      IF(RSAMMESIDPOS(JJ) .GE. 1) THEN
        IF(DEBUG .GE. 1) THEN
          WRITE(16,*)'SAM SITE #',JJ,' FINAL MESSAGE POINTER=',
     &      RSAMMESIDPOS(JJ)
        ENDIF
      ENDIF
  100 CONTINUE
      WRITE(16,*) '-----------------------------------------------'


      RETURN
      END
--------------------------------
      SUBROUTINE SAMOFF

C.....LATEST CHANGE MADE 6/28/91 BFS.....
C     TURNS OFF A SAM SITE AT A GIVEN TIME DURING THE SIMULATION.
C     USED TO TEST THE IADS MODIFICATIONS.
C
C     CALLED BY - IADS
C     CALLS     - HOLDMES
C.....

      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      REAL MSG(13)

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG
      REAL TIME
C.....COMMON BLOCK.....

      INCLUDE 'MESID DEF'

      COMMON/DEBUG/DEBUG
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                                  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
```

```
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....VARIABLES IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     TIME - CURRENT SIM. TIME
C.....LOCAL TO THIS SUBROUTINE.....
C     MSG(13) - HOLDS DATA CONTAINING MESSAGE TO BE SENT
C.....START PROGRAM.....

C     WRITE(*,*) 'SAMOFF'

C ... ADWOC SENDS MESSAGE TO SAM SITE #5 TO TURN OFF IN 10 SEC ...

      MSG(1)=TIME
      MSG(2)=TIME+10.0
      MSG(3)=1.0
      MSG(4)=3.0
      MSG(5)=1.0
      MSG(6)=5.0
      MSG(7)=5.0
      MSG(8)=0.0
      MSG(9)=REAL(MSGID)
      MSG(10)=3.0
      MSG(11)=1.0
      MSG(12)=5.0
      MSG(13)=5.0

      IF (DEBUG .GE. 1) THEN
        WRITE(16,1003) MSG(1),MSG(2),NINT(MSG(7))
 1003   FORMAT(F5.0,' SEC ADWOC WILL SEND SHUT OFF MESSAGE AT TIME',
     &  F5.0,' TO SAM SITE # ',I2)
        WRITE(16,*) '         MSGID=',NINT(MSG(9))
      ENDIF

      CALL HOLDMES(MSG)
      MSGID=MSGID+1

      RETURN
      END
---------------------------------
      SUBROUTINE SAMON

C.....LATEST CHANGE MADE 6/28/91 BFS.....
C     TURNS ON A SAM SITE AT A GIVEN TIME DURING THE SIMULATION.
C     USED TO TEST THE IADS MODIFICATIONS.
C
C     CALLED BY - IADS
C     CALLS     - HOLDMES
C.....
```

```
      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      REAL MSG(13)

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      INTEGER DEBUG
      REAL TIME
C.....COMMON BLOCK.....

      INCLUDE 'MESID DEF'

      COMMON/DEBUG/DEBUG
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)

C..... VARIABLES USED IN THIS SUBROUTINE.....
C.....                                .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....VARIABLES IN COMMON BLOCK.....
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     TIME - CURRENT SIM. TIME
C.....LOCAL TO THIS SUBROUTINE.....
C     MSG(13) - HOLDS DATA CONTAINING MESSAGE TO BE SENT
C.....START PROGRAM.....

C     WRITE(*,*) 'SAMOFF'

C ... ADWOC SENDS MESSAGE TO SAM SITE #5 TO TURN OFF IN 10 SEC ...

      MSG(1)=TIME
      MSG(2)=TIME+10.0
      MSG(3)=1.0
      MSG(4)=3.0
      MSG(5)=1.0
      MSG(6)=5.0
      MSG(7)=5.0
      MSG(8)=1.0
      MSG(9)=REAL(MSGID)
      MSG(10)=3.0
      MSG(11)=1.0
      MSG(12)=5.0
```

```
      MSG(13)=5.0

      IF (DEBUG .GE. 1) THEN
        WRITE(16,1003) MSG(1),MSG(2),NINT(MSG(7))
 1003   FORMAT(F5.0,' SEC ADWOC WILL SEND SHUT OFF MESSAGE AT TIME',
     &  F5.0,' TO SAM SITE # ',I2)
        WRITE(16,*) '         MSGID=',NINT(MSG(9))
      ENDIF

      CALL HOLDMES(MSG)
      MSGID=MSGID+1

      RETURN
      END
--------------------------------
      SUBROUTINE REDIN


C.....***.....LATEST CHANGE MADE 6/25/91 BFS.....

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      INTEGER BOMBS,IDHAC,II,JJ,K,KK,L,LL,N,NCK,NO0,NO1,NPENTYPE

      CHARACTER TITLE*70

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      REAL*8 ALTL,ALTU,CON1,CON2,R,SITED1,SITED2,SITED3,SITED4,SITED5

      REAL ALTARM,ALTMN,ANGM,BM,BOMPK,CKON1,CS,C2TIME,DIST,DT,
     $ESITE1,ESITE2,FEND,FTPRTA,GDT,GTFUS,HT,MWAITIM,OWAITIM,
     $PENTARPK,PKM,PPK,PS,PSAMIN,PSSMIN,RNGARM,SSEND,SIG,T,TARM,THETJ,
     $TLB,TLDRM,TRACK,V,VM,X,XS,Y,YS,Z,ZS

      INTEGER BOM,BOMPT,BOMTGT,CNTRL2,C2END,C2START,C2WPN,DEBUG,
     $DRMTG,I,IOP,ISECT,J,LNCH1,LNCH2,LMAXS,L1CKPT,L2CKPT,L1WPN,L2WPN,
     $MDEST,MSITE,NAC,NCKPT,NHRMAC,NUMB,NUMD,NUMR,NSITE,NUMSHT,ORDR,
     $PENTYPE,PMPREF,TMEM,MAXMS,MSID,NTARGETS

C     REAL*8 ISEED
      REAL ISEED
C  ====  ABOVE MODIFICATIONS MADE FOR MAINFRAME VERSION ONLY
==========

C.....COMMON BLOCK.....
```

```
      COMMON/ARMD/NUMB(IM),NUMR(IM),TLB(IM,8),NPP,NRE,NHRMAC,
     $NDP,NUMD(IM)
      COMMON/ARM1/A(48),B(48),PL(48),TL(48),XL(48),YL(48),JPM(48),
     $TPM(48),ALTMN
      COMMON/DEBUG/DEBUG
      COMMON/CRUISE/IOP(10),TIMEL,RRAD(JM),IT(JM)
      COMMON/DBLP/ISEED
      COMMON/ENV/ESITE1(9,10),ESITE2(9,10)
      COMMON/FOTPNT/FEND(7,2),SSEND(7,2),FTPRTA(26,7),FE,SE
      COMMON/INDATA/PPK(JM)
      COMMON/NEW/BM(8),PMPREF(IM,8),DRMTG(IM,8),TLDRM(IM,8),NUMR1(IM),
     $TMEM,ANGM,ORDR
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/RCTIME/TRACK(9),TTRACK(JM),NUMSHT(JM),NOSHOT(JM),NSHOT(JM)
      COMMON/SHOTA/XXINT(JM),IX(JM),MASKTIM(JM),MWAITIM(JM),OUTIM(JM),
     $OWAITIM(JM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)
      COMMON/SITG/SITED1(9,9),SITED2(9,9),SITED3(9,9),SITED4(9,9),
     $SITED5(9,9),ALTL(9,JM),ALTU(9,JM),CON1(9,JM),CON2(9,JM),R(9,JM),
     $LMAXS(9)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)
      COMMON/TERA/HT(JM),DIST(JM),CS(IM),SIG(IM)
      COMMON/TRAJ/TARM(10),ALTARM(10),RNGARM(10),XHBA(320),ZHBA(320)
      COMMON/WEAPONS/BOM(IM),BOMPT(IM,4),BOMTGT(IM,2,4),BOMTIM(IM,4),
     $BOMPK(2,20),NTARGETS,PSAL(IM,2,4),LNCH1(IM),LNCH2(IM),
     $L1CKPT(IM,4),L2CKPT(IM,4),L1WPN(IM,4),L2WPN(IM,4),LNCH1TIM(IM,4),
     $CNTRL2(IM),C2ETIM(IM,4),C2START(IM,4),C2END(IM,4),C2WPN(IM,4),
     $C2TIME(IM,4),LNCH2TIM(IM,4),C2STIM(IM,4),PSAC(IM,4),WHO(IM,2,2,4),
     $C2DEDTIM(IM)
      COMMON/MUL/MAXMS,MSID(15,2)

      INCLUDE 'QIADS DEF'

C.....DESCRIPTION OF VARIABLES IN THIS SUBROUTINE.....
C.....                                  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....VARIABLES IN COMMON BLOCK USED IN THIS SUBROUTINE .....
C     ALTARM(10) - ALTITUDE (IN NMI) OF ARM AT GIVEN TIMES FROM LAUNCH.
C                - THIS VARIABLE ALONG WITH TARM(10) AND RNGARM(10)
C                - FORM THE BASIC ARM TRAJECTORY AND THE DATA IS GIVEN
C                - IN THE INPUT DATAFILE
C     ALTMN - ALTITUDE BELOW WHICH PARM STARTS SEARCHING FOR A SITE
```

```
C   ANGM - HALF ANGLE FIELD OF VIEW OF ARM
C   BM(8)
C   BOM(IM) - MAX # OF BOMB GROUPS THAT A/C IM WILL DROP
C   BOMPK(2,20) - BOMB PK AGAINST SAM AND NON-TARGETS
C   BOMPT(IM,N) - CHECKPOINT # WHERE A/C IM DROPS BOMB GROUP N
C   BOMTGT(IM,2,N) - TARGET TYPE AND TARGETS LOCATION # FOR BOMBS N
C   CKON1(JM)
C   CNTRL2(IM) - MAX # OF TYPE 2 STANDOFF WEAPONS THAT A/C IM CONTROLS
C   CS(IM)
C   C2END(IM,N) - CHECKPOINT # WHERE CONTROL A/C IM ENDS CONTROL OF
C            - ITS TYPE 2 STANDOFF WEAPON # N
C   C2START(IM,N) - CHECKPOINT # WHERE A/C IM STARTS CONTROLING TYPE
C             - 2 STANDOFF WEAPON # N
C   C2TIME(IM,N) - TIME (IN SECONDS) BEFORE IMPACT WHEN CONTROL A/C IM
C             - MAY BE KILLED AND ITS TYPE 2 STANDOFF WEAPON # N IT
C             - WAS CONTROLLING MAY STILL HIT THE TARGET
C   C2WPN(IM,N) - PENETRATOR # FOR TYPE 2 STANDOFF WEAPON # N THAT
C            - WILL BE CONTROLLED BY A/C IM
C   DEBUG - FLAG FOR DEBUG PRINTOUTS
C   DIST(JM)
C   DRMTG(IM,8)
C   DT - TACOPS TIME INCREMENT (1 SECOND)
C   ESITE1(9,10) - LAUNCH BOUNDRY RANGE VALUES FOR SAM SITES
C   ESITE2(9,10) - LAUNCH BOUNDRY ALTITUDE VALUES FOR SAM SITES
C   FEND(7,2)
C   FTPRTA(26,7)
C   GDT(JM) - KILL ASSES DELAY TIME FOR SAM SITE JM
C   GTFUS(JM)
C   HT(JM)
C   I - PEN. LOOP COUNTER
C   IOP(10) - HOLDS FLAGS(1/0) TO SELECT PROGRAM RUNNING MODES
C   ISECT(IM,JM) - HOLDS A FLAG(1/0) FOR EACH PENETRATOR IM WHICH
C             - INDICATES WHETHER EACH SAM SITE JM CAN(=1) OR
C             - CANNOT(=0) FIRE AT THIS PENETRATOR
C   ISEED - DOUBLE PRECISION # USED AS SEED IN RANDOM # SUB. DRAND
C   J - SAM SITE LOOP COUNTER
C   LNCH1(IM) - MAX # OF TYPE 1 STANDOFF WEAPONS AC/ IM LAUNCHES
C   LNCH2(IM) - MAX # OF TYPE 2 STANDOFF WEAPONS AC/ IM LAUNCHES
C   LMAXS(9)
C   L1CKPT(IM,N) - CHECKPOINT # WHERE A/C IM LAUNCHES ITS TYPE 1
C             - STANDOFF WEAPON # N
C   L2CKPT(IM,N) - CHECKPOINT # WHERE A/C IM LAUNCHES ITS TYPE 2
C             - STANDOFF WEAPON # N
C   L1WPN(IM,N) - PENETRATOR # FOR TYPE 1 STANDOFF WEAPON # N THAT
C            - A/C IM WILL LAUNCH
C   L2WPN(IM,N) - PENETRATOR # FOR TYPE 2 STANDOFF WEAPON THAT
C            - A/C IM WILL LAUNCH
C   MDEST(IM,2) - FOR PENETRATOR IM;
C            - IF MDEST(IM,1)=1; PENETRATOR IS TARGETTED AGAINST
C            - A SAM SITE AND MDEST(IM,2) HOLDS THE SITE #.
```

```
C            - IF MDEST(IM,1)=2; PENETRATOR IS TARGETTED AGAINST
C            - A NON-SAM SITE TYEP TARGET AND MDEST(IM,2) HOLDS
C            - THE TARGET #.
C    MSITE(JM) - HOLDS THE SAM SITE TYPE (1-9) FOR SITE JM
C    MWAITIM(J) - MAXIMUM WAIT TIME (IN SECONDS) AFTER THE TARGET IS
C           - TERRAIN MASKED BEFORE THIS SITE TYPE LOSES TRACK AND
C           - THE OUTBOUND SAM IS LOST (DATA READ IN AS SITE TYPE
C           - AND IS CONVERTED TO SITE LOCATION # IN SUB INIT)
C    NAC - MAX # OF PENETRATORS
C    NCKPT(IM)
C    NHRMAC - TOTAL NUMBER OF A/C CARRYING ARM MISSILES
C    NUMB(IM) - # OF PREPROGRAMMED ARM MISSILES ON A/C IM
C    NUMD(IM) - # OF DIRECTIVE ARM MISSILES ON A/C IM
C    NUMR(IM) - # OF REACTIVE ARM MISSILES ON A/C IM
C    NSITE - MAX # OF SAM SITES
C    NUMSHT(JM) - MAX # OF SALVOS THIS SAM TYPE HAS AVAILABLE
C    ORDR
C    OWAITIM(J) - MAXIMUM WAIT TIME (IN SECONDS) AFTER THE TARGET IS
C           - OUTSIDE OF THE SITE LAUNCH ENVELOPE BEFORE THIS SITE
C           - TYPE LOSES TRACK AND THE OUTBOUND SAM IS LOST (DATA
C           - READ IN AS SITE TYPE AND IS CONVERTED TO SITE
C           - LOCATION # IN SUB INIT)
C    PENTARPK(2,2,20) - PK OF EACH PENETRATOR TYPE FOR EACH SAM SITE
C                  - TYPE OR EACH NON-AD TARGET NUMBER
C    PENTYPE(IM) - INTEGER INDICATING THE PENETRATOR TYPE
C    PKM(9) - PK OF ARM FOR THE 9 SAM TYPES
C    PMPREF(IM,8)
C    PPK(JM)
C    PS(JM) - INITIAL PS OF EACH SAM SITE
C    PSAMIN - MINIMUM SURVIVABILITY OF PENETRATOR;IF PSA(IM) IS BELOW
C         - THIS, THE PENETRATOR IS REMOVED FROM THE SIMULATION
C    PSSMIN - MIN SURVIVAL PROBABILITY THAT THE SAM SITE MUST MAINTAIN
C         - IN ORDER TO STAY IN THE SIMULATION
C    RNGARM(10) - DOWNRANGE (IN NMI) OF ARM FROM LAUNCH.
C           - THIS VARIABLE ALONG WITH TARM(10) AND ALTARM(10)
C           - FORM THE BASIC ARM TRAJECTORY AND THE DATA IS GIVEN
C           - IN THE INPUT DATAFILE
C    SSEND(7,2)
C    SIG(IM)
C    T(IM,KM)
C    TARM(10) - TIME (IN SECONDS) FROM LAUNCH OF ARM;READ IN FROM THE
C           - INPUT DATAFILE. ALONG WITH ALTARM(10) AND RNGARM(10),
C           - THIS FORMS THE BASIC ARM TRAJECTORY FROM WHICH THE
C           - EXACT ARM TRAJECTORY (FOR EACH SECOND AFTER LAUNCH)
C           - IS COMPUTED.
C    THETJ(JM)
C    TLB(IM,8)
C    TLDRM(IM,8)
C    TMEM - MAX TIME (IN SECONDS) DURING WHICH THE ARM WILL STILL HIT
C        - THE SAM SITE EVEN THOUGH THAT SITE HAS STOPPED RADIATING.
```

```
C     TRACK(9)
C     V(IM,KM)
C     VM(JM)
C     X(IM,KM)
C     XS(JM) - X POSITION (IN NMI) OF SAM SITE JM
C     Y(IM,KM)
C     YS(JM) - Y POSITION (IN NMI) OF SAM SITE JM
C     Z(IM,KM)
C     ZS(JM) - Z POSITION (IN NMI) OF SAM SITE JM
C.....VARIABLES LOCAL TO THIS SUBROUTINE.....
C     BOMBS - FLAG (1/0) TO INDICATE IF ANY A/C ARE TO DROP BOMBS
C     IDHAC
C     II
C     JJ
C     K
C     KK
C     L
C     LL
C     N
C     NCK
C     NO0
C     NO1
C     NPENTYPE - MAX # OF DIFFERENT PENETRATOR TYPES USED IN THE DATA
C     NTARGETS
C     TITLE
C.....START PROGRAM.....

C     WRITE(*,*) 'REDIN'

C     OPEN(5,FILE='A:ZTACOPS.IN',STATUS='OLD')
C     OPEN(7,FILE='A:ZTACOPS.DOC',STATUS='NEW')
      OPEN(15,FILE='/TACOPS IN')
      OPEN(17,FILE='/TACOPS DOC B')


      WRITE(*,*) 'DO YOU WANT ANY DEBUG OUTPUT?(0=NO; 1=YES,SOME; 2=YES,
     $ MORE; 3=YES, ALL)'
      READ(*,*) DEBUG
C     IF (DEBUG .GE. 1) OPEN(6,FILE='A:ZTDEBUG.OUT',STATUS='NEW')
      IF (DEBUG .GE. 1) OPEN(16,FILE='/TDEBUG OUT B')

      DT=1.0
      BOMBS=0
C.....START READING INPUT DATA.....

      READ(15,1000) TITLE
 1000 FORMAT(A70)
      IF (DEBUG .GE. 1)  WRITE(16,1000) TITLE

C.....WRITE FIRST DATA RECORD TO DOC FILE.....
```

```
      WRITE(17,3001) TITLE
 3001 FORMAT(A70)

C.....

      READ (15,*) (IOP(II),II=1,10)

C.....IOP(1)=0 EXPECTED VALUE: IOP(1)=1 STOCHASTIC MODEL.....
C.....IOP(2)=0 REG DOCTRINE: IOP(2)=1 PERFECT COODINATION.....
C.....IOP(3)=0 FREE FIRING : IOP(3)=1 1-ON-1 FIRING DOCTRINE.....
C.....THE FOLLOWING ARE BUILT IN INPUTS.....

      ISEED=1000000.00

      IF (IOP(1) .EQ. 1) THEN
C       OPEN(3,FILE='A:TIMESEED',STATUS='OLD')
        OPEN(13,FILE='/TIMESEED')
        READ(13,*) ISEED
        CLOSE (13)
      ENDIF


      IF (IOP(9) .EQ. 1)  WRITE(18,1000) TITLE

C.....READ AIRCRAFT DATA.....

      READ (15,*) NAC,NSITE,PSAMIN,NPENTYPE,NTARGETS

C.....WRITE SECOND DATA RECORD TO DOC FILE.....

      WRITE(17,3002) NAC,NSITE,NTARGETS
 3002 FORMAT(I3,I3,I3)

C.....*** READ IN TOTAL # AND ID # OF MULTITARGETTING SAMS.....

      READ(15,*) MAXMS
      IF (MAXMS .EQ. 0) GOTO 15
      DO 10 J=1,MAXMS
      READ(15,*) MSID(J,1),MSID(J,2)
   10 CONTINUE
   15 CONTINUE
C.....READ SAM SITE DATA.....

      DO 20 J=1,NSITE
   20 READ(15,*) XS(J),YS(J),ZS(J),VM(J),GDT(J),PPK(J),MSITE(J),
     $THETJ(J),CKON1(J),GTFUS(J),PS(J),NUMSHT(J)

C.....READ PENETRATOR TYPE AND TARGET DATA.....

      DO 30 I=1,NAC
```

```
      READ(15,*) PENTYPE(I)

C.....FOR PENETRATOR TYPE 0 .....


      IF (PENTYPE(I) .EQ. 0) THEN

        READ(15,*) BOM(I),LNCH1(I),LNCH2(I),CNTRL2(I)

        IF (BOM(I) .GE. 1) THEN
          DO 300 N=1,BOM(I)
          READ(15,*) BOMPT(I,N),BOMTGT(I,1,N),BOMTGT(I,2,N)
          BOMBS=1
  300      CONTINUE
        ENDIF

        IF (LNCH1(I) .GE. 1) THEN
          DO 310 N=1,LNCH1(I)
          READ(15,*) L1CKPT(I,N),L1WPN(I,N)
  310      CONTINUE
        ENDIF

        IF (LNCH2(I) .GE. 1) THEN
          DO 320 N=1,LNCH2(I)
          READ(15,*) L2CKPT(I,N),L2WPN(I,N)
  320      CONTINUE
        ENDIF

        IF (CNTRL2(I) .GE. 1) THEN
          DO 330 N=1,CNTRL2(I)
          READ(15,*) C2START(I,N),C2END(I,N),C2WPN(I,N),C2TIME(I,N)
  330      CONTINUE
        ENDIF


        READ(15,*) NCKPT(I),T(I,1)
      ENDIF


C.....FOR PENETRATOR TYPE 1 .....

      IF (PENTYPE(I) .EQ. 1) THEN
        READ(15,*) MDEST(I,1),MDEST(I,2)
        READ(15,*) NCKPT(I)
      ENDIF

C.....FOR PENETRATOR TYPE 2 .....

      IF (PENTYPE(I) .EQ. 2) THEN
        READ(15,*) MDEST(I,1),MDEST(I,2)
```

```
      READ(15,*) NCKPT(I)
    ENDIF


C.....READ CHECKPOINT POSITION AND VELOCITY DATA.....

    NCK=NCKPT(I)
    DO 40 K=1,NCK
 40 READ (15,*) X(I,K),Y(I,K),Z(I,K),V(I,K)
 30 CONTINUE

C.....READ THE PK OF THE WEAPONS AGAINST THE TARGETS.....

    IF (NPENTYPE .EQ. 0) GOTO 53

    DO 50 II=1,NPENTYPE
    READ(15,*) (PENTARPK(II,1,L),L=1,9)
    IF (NTARGETS .GT. 0) READ(15,*) (PENTARPK(II,2,L),L=1,NTARGETS)
 50 CONTINUE

 53 CONTINUE


    IF (BOMBS .EQ. 0) GOTO 55

    READ(15,*) (BOMPK(1,L),L=1,9)
    IF (NTARGETS .GT. 0) READ(15,*) (BOMPK(2,L),L=1,NTARGETS)

 55 CONTINUE



    DO 60 I=1,NAC
    READ(15,*)  (ISECT(I,J),J=1,NSITE)
 60 CONTINUE

C.....READ THE ARM AIRCRAFT DATA.....

    READ (15,*) NHRMAC,TMEM


    IF(NHRMAC .EQ. 0) GO TO 140

    READ(15,*) (PKM(L),L=1,9)

    DO 70 II=1,NHRMAC
    READ (15,*) IDHAC,NUMD(IDHAC),NUMB(IDHAC),NUMR(IDHAC)
    IF(NUMD(IDHAC) .EQ. 0) GO TO 90
    NO0=NUMD(IDHAC)
```

```
      DO 80 JJ=1,NO0
   80 READ (15,*) DRMTG(IDHAC,JJ),TLDRM(IDHAC,JJ)

   90 IF(NUMB(IDHAC) .EQ. 0) GO TO 70
      NO1=NUMB(IDHAC)

      DO 100 JJ=1,NO1
  100 READ(15,*) TLB(IDHAC,JJ),PMPREF(IDHAC,JJ)


   70  CONTINUE

C.....READ DEFENSE SITE DATA.....

  140  READ(15,*) PSSMIN

      DO 150 J=1,9
      READ (15,*) LL,TRACK(J),MWAITIM(J),OWAITIM(J)
      LMAXS(J)=LL-1

      DO 160 L=1,LL
      READ (15,*) ESITE1(J,L),ESITE2(J,L)
  160  CONTINUE

  150  CONTINUE

C.....READ ARM FOOTPRINTS.....

      READ(15,*)  (FEND(L,1),L=1,7)

      READ(15,*)  (SSEND(L,1),L=1,7)

      READ (15,*) KK

      DO 190 K=1,KK
      READ (15,*) (FTPRTA(K,L),L=1,7)
   190 CONTINUE

C.....READ ARM TRAJECTORY.....

      READ(15,*) ANGM,ALTMN

      DO 210 II=1,10
   210 READ(15,*) TARM(II),ALTARM(II),RNGARM(II)

      READ(15,*) ORDR
      READ(15,*) (BM(L),L=1,ORDR+1)


C.....READ TERRRAIN AND AIRCRAFT RADAR DATA.....
```

```
      DO 230 JJ=1,NSITE
      READ(15,*) HT(JJ),DIST(JJ)
  230 CONTINUE

      DO 240 I=1,NAC
      READ(15,*) CS(I),SIG(I)
  240 CONTINUE

      READ (15,*) NEWSITES

C ... IF NOT IADS AND NO EW SITES, YOU ARE DONE READING .....
      IF (IOP(3) .LE. 4  .AND.  NEWSITES .EQ. 0) GOTO 900

C ... IF IADS OR EW SITES EXIST, DO MORE READING IN REDIAD SUB.....
      CALL REDIAD


  900 CLOSE(15)

      CALL ENVLOP

      RETURN
      END
----------------------------------
      PROGRAM TACOPS


C.............LATEST CHANGES MADE ON 6/27/91 BFS.....

      PARAMETER (IM=30,JM=25,KM=20)

C     IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      REAL AJ,INITDAMG,SDAMG,TGTDAMG
      INTEGER ICO,IDAT,KT,L,LIM,LLOOP,LOOP,LP
      CHARACTER DATS*36,D4S*5

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      REAL ADAMG,DELTI,DMDAMG,DT,ITM,PMDAMG,PS,PSA,PSAMIN,PST,RMDAMG,
     $TDAMG,TIME,TIMEL,XS,YS
      INTEGER DEBUG,I,IOP,J,MDEST,MSITE,NAC,NDP,NHRMAC,NOSHOT,NPP,
     $NRE,NSHOT,NSITE,NTARGETS

C.....COMMON BLOCK.....

      COMMON/ARMD/NUMB(IM),NUMR(IM),TLB(IM,8),NPP,NRE,NHRMAC,
     $NDP,NUMD(IM)
```

```
      COMMON/CRUISE/IOP(10),TIMEL,RRAD(JM),IT(JM)
      COMMON/DAMAGE/RMDAMG,PMDAMG,DMDAMG,ADAMG,TDAMG,PMFLG(48),
     $ISTUS(48),ITM(48),DELTI
      COMMON/DEBUG/DEBUG
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/RCTIME/TRACK(9),TTRACK(JM),NUMSHT(JM),NOSHOT(JM),NSHOT(JM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)
      COMMON/WEAPONS/BOM(IM),BOMPT(IM,4),BOMTGT(IM,2,4),BOMTIM(IM,4),
     $BOMPK(2,20),NTARGETS,PSAL(IM,2,4),LNCH1(IM),LNCH2(IM),
     $L1CKPT(IM,4),L2CKPT(IM,4),L1WPN(IM,4),L2WPN(IM,4),LNCH1TIM(IM,4),
     $CNTRL2(IM),C2ETIM(IM,4),C2START(IM,4),C2END(IM,4),C2WPN(IM,4),
     $C2TIME(IM,4),LNCH2TIM(IM,4),C2STIM(IM,4),PSAC(IM,4),WHO(IM,2,2,4),
     $C2DEDTIM(IM)

C.....DESCRIPTION OF VARIABLES IN THIS SUBROUTINE.....
C.....  .....
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....VARIABLES IN COMMON BLOCK USED IN THIS SUBROUTINE .....
C     ADAMG - TOTAL HARDKILL MISSILE DAMAGE BY SAM SYSTEMS
C     DEBUG - FLAG FOR DEBUG PRINTOUTS
C     DELTI - DELAY TIME (IN SECONDS) TO READY REACTIVE ARM
C     DMDAMG - TOTAL SAM SITE DAMAGE BY DIRECTED ARMS
C     DT - TACOPS TIME INCREMENT (1 SECOND)
C     I - PEN. LOOP COUNTER
C     IOP(10) - HOLDS FLAGS(1/0) TO SELECT PROGRAM RUNNING MODES
C     J - SAM SITE LOOP COUNTER
C     MDEST(IM,2) - FOR PENETRATOR IM;
C               - IF MDEST(IM,1)=1; PENETRATOR IS TARGETTED AGAINST
C               - A SAM SITE AND MDEST(IM,2) HOLDS THE SITE #.
C               - IF MDEST(IM,1)=2; PENETRATOR IS TARGETTED AGAINST
C               - A NON-SAM SITE TYEP TARGET AND MDEST(IM,2) HOLDS
C               - THE TARGET #.
C     MSITE(JM) - HOLDS THE SAM SITE TYPE (1-9) FOR SITE JM
C     NAC - MAX # OF PENETRATORS
C     NDP - RUNNING TOTAL OF DARMS FIRED SO FAR
C     NHRMAC - TOTAL NUMBER OF A/C CARRYING ARM MISSILES
C     NOSHOT(JM) - MAX # OF SALVOS AVAILABLE TO SAM SITE (JM)
C     NPP - RUNNING TOTAL # OF PREPROGRAMMED ARMS FIRED
C     NRE - RUNNING TOTAL # OF REACTIVE ARMS FIRED
C     NSHOT(JM) - RUNNING TOTAL # OF SALVOS SAM SITE JM HAS FIRED
C     NSITE - MAX # OF SAM SITES
```

```
C     NTARGETS - MAX # OF NON-SAM TARGETS IN THE DATAFILE
C     PMDAMG - TOTAL SAM SITE DAMAGE BY PREPROGRAMMED ARMS
C     PS(JM) - SURVIVAL PROBABILITY OF SAM SITE JM
C     PSA(IM) - SURVIVAL PROBABILITY OF PENETRATOR IM
C     PSAMIN - MINIMUM SURVIVABILITY OF PENETRATOR;IF PSA(IM) IS BELOW
C           - THIS, THE PENETRATOR IS REMOVED FROM THE SIMULATION
C     PST(20) - PS OF NON-AD TYPE TARGETS
C     RMDAMG - TOTAL SAM SITE DAMAGE BY REACTIVE ARMS
C     TDAMG - TOTAL SAM SITE DAMAGE BY ALL WEAPONS
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C     TIMEL - LATEST TIME (IN SECONDS) WHEN THE SLOWEST PENETRATOR
C          - WILL BE AT ITS LAST CHECKPOINT
C     XS(JM) - X POSITION (IN NMI) OF SAM SITE JM
C     YS(JM) - Y POSITION (IN NMI) OF SAM SITE JM
C.....VARIABLES LOCAL TO THIS SUBROUTINE.....
C     AJ
C     D4S
C     ICO
C     IDAT
C     INITDAMG - INITIAL TOTAL SITE DAMAGE AT THE START OF THE RUN
C     KT
C     L
C     LIM
C     LLOOP
C     LOOP
C     LP - COUNTER FOR LOOPS DONE IN A STOCHASTIC MODE RUN
C     SDAMG - TOTAL SAM SITE DAMAGE BY HARDKILL MISSILES
C     TGTDAMG - TOTAL NON-SAM TYPE GROUND TARGET DAMAGE BY WEAPONS
C.....START PROGRAM.....

C     WRITE(*,*) 'TACOPS'

      WRITE(*,*) ' '
      WRITE(*,*) '              TACOPS - MOD 10   '
      WRITE(*,*) ' '
C       WITH UP TO 60 PENETRSTORS AND 15 CHECKPOINTS EACH,
C       30 SAM SITES/20 NON-SAM TARGETS AND 48 ARMS.
C
C       LATEST MODS INCLUDE THE CAPABILITY TO:
C
C       1) HAVE ATTRITION ON NON-SAM TYPE TARGETS
C       2) HAVE MORE THAN ONE STANDOFF WEAPON TYPE WITH
C          UNIQUE PK FOR THE SAM AND NON-SAM TYPE TARGETS
C       3) HAVE EACH SAM SITE WITH A UNIQUE ALTITUDE
C       4) HAVE AN ARM PK FOR EACH UNIQUE SAM TYPE
C       5) ALLOW A/C TO DROP BOMBS AND LAUNCH PENETRATORS
C          IF THEY SURVIVE TO THE WEAPONS RELEASE POINT
C       6) HAVE EACH SITE WITH A UNIQUE STARTING PS AND
C          NUMBER OF AVAILABLE SALVOS
C       7) CORRECTION TO PARM FLIGHT PATH COMPUTATIONS; SUB FUTLOK
```

```
C       REPLACES SUB LOCKON - WILL NOT ALLOW SITES TO FIRE IF, AT
C       INTERCEPT, THE FUZE ANGLE WILL BE BAD; EXPANDED DOCUMENTING
C       OUTPUT DATA FOR ZTR PROGRAM IN THE AREAS OF SAM SITE
C       RADIATING STATUS AND ARM FUNCTIONS; DEFINED TYPE 1 STANDOFF
C       WEAPON TO BE A FIRE AND FORGET TYPE, AND A NEW TYPE 2
C       STANDOFF WEAPON THAT REQUIRES POST-LAUNCH CONTROL OR
C       DESIGNATION
C     8) CORRECTED CODE IN SHOT SUBROUTINES TO NOT ALLOW SITES TO
C       FIRE AND INTERCEPT PENETRATORS THAT ARE OUT OF ENVELOPE OR
C       ARE TERRAIN MASKED; ADDED TO INPUT DATAFILE VALUES FOR THE
C       MAXIMUM WAIT TIME BEFORE THE SITE LOSES THE TRACK FOR TERRAIN
C       MASKING AND THE OUT OF ENVELOPE CONDITION WHILE A SAM IS
C       ENROUTE

      DELTI=5.
      ICO=0
      LLOOP=0

      CALL VARIBL
      CALL REDIN


      INITDAMG=0.0

      DO 70 J=1,NSITE
      INITDAMG=INITDAMG+1.0-PS(J)
   70 CONTINUE


      LOOP=IOP(10)

      IF (IOP(9) .NE. 1) GO TO 5
C     OPEN(8,FILE='A:ZTACOPS.OUT',STATUS='NEW')
C     REWIND 8
      OPEN(18,FILE = '/TACOPS OUT')
      REWIND 18


    5 DO 10 LP=1,LOOP

      AJ=0.0
      CALL INIT(LLOOP,SDAMG)
      LIM=NINT((TIMEL-TIME)/DT+1.0)



      DO 20 KT=1,LIM

      CALL TICK2(SDAMG)
```

```
C.....SHOW PRESENT AND ENDING TIME ON SCREEN....

C     WRITE(*,1999) TIME,LIM
C1999 FORMAT('+PRESENT TIME = ',F5.0,'  TIME PROGRAM ENDS= ',I4)

      IF(IOP(3) .LE. 4) THEN
        CALL SHOT
      ELSE
        CALL IADS
      ENDIF

      IF(NHRMAC .EQ. 0) GO TO 30

C= = = = = = = PC VERSION WAS 'CALL ARM' = = = = = = = = = = =
      CALL AARM

      IF(NDP .GT. 0) CALL DARM
      IF(NPP .GT. 0) CALL PARM(AJ)
      IF(NRE .GT. 0) CALL RARM

   30 CONTINUE


      DO 40 I=1,NAC
   40 IF (PSA(I) .GE. PSAMIN) GO TO 20

      ICO=ICO+1
      IF(ICO .LT. 2) GO TO 20

      IF (DEBUG .GE. 1) THEN
        WRITE (16,1008)
 1008   FORMAT(' ALL PENETRATORS SHOT DOWN!')
        WRITE (*,*) ' '
        WRITE (*,*) ' ALL PENETRATORS SHOT DOWN!'
      ENDIF

C.....WRITE EVENT AND DATA TO DOC FILE.....

      WRITE(17,2001) 35,2
 2001 FORMAT(I2,I2)
      WRITE(17,3001) ' ALL PENETRATORS SHOT DOWN!'
 3001 FORMAT(A30)

      GO TO 80

   20 CONTINUE

      WRITE (*,1009)
 1009 FORMAT(' ALL GIVEN CHECKPOINTS VISITED.')
```

```
C.....WRITE EVENT AND DATA TO DOC FILE.....

      WRITE(17,2002) 36,2
 2002 FORMAT(I2,I2)
      WRITE(17,3002) ' ALL CHECKPOINTS VISITED.'
 3002 FORMAT(A30)

   80 IF (DEBUG .GE. 1)  WRITE(16,*) '--'



      WRITE(17,2014) 58,1
 2014 FORMAT(I2,I2)
      WRITE(17,3014) 0.0,0,NDP,NPP,NRE,0.0
 3014 FORMAT(F5.0,I2,I2,I3,I3,F9.4)

C.....PRINT OUT SAM SITES FINAL PS.....


      TDAMG=0.

      DO 50 J=1,NSITE
      D4S='     '
      IDAT=MAX0(0,NOSHOT(J)-NSHOT(J))
      TDAMG=TDAMG+1.0-PS(J)

      IF (DEBUG .GE. 1) THEN
        WRITE(16,1000) D4S,J,IDAT,NOSHOT(J),PS(J),XS(J),YS(J)
 1000   FORMAT(A5,' SITE ',I2,' HAS ',I3,' SALVOS LEFT OF ',I3,' PS ',
     $  F6.4,1X,F5.1,1X,F5.1)
      ENDIF

      WRITE(17,2003) 37,1
 2003 FORMAT(I2,I2)
      WRITE(17,3003) 0.0,0,J,IDAT,NOSHOT(J),PS(J)
 3003 FORMAT(F5.0,I2,I2,I3,I3,F9.4)

   50 CONTINUE


      TDAMG=TDAMG-INITDAMG

      IF (DEBUG .GE. 1) THEN
        WRITE(16,*) ' '
        WRITE(16,1001) INITDAMG
 1001   FORMAT(' TOTAL SITE DAMAGE BEFORE THIS RUN =  ',F7.3)
        WRITE(16,1002) TDAMG+INITDAMG
 1002   FORMAT(' TOTAL SITE DAMAGE TO THE PRESENT  =  ',F7.3)
        WRITE(16,1003) TDAMG
 1003   FORMAT(' TOTAL SITE DAMAGE DURING THIS RUN =  ',F7.3)
```

```
        WRITE(16,*) ' '
      ENDIF

      IF (DEBUG .GE. 1) THEN
        DATS='        AIRCRAFT DAMAGE BY SAMS = '
        WRITE(16,1007) DATS,ADAMG
        DATS=' SITE DAMAGE BY NON-ARM WEAPONS = '
        WRITE(16,1007) DATS,SDAMG
C       DATS='   SITE DAMAGE BY DIRECTED ARMS = '
C       WRITE(16,1007) DATS,DMDAMG
        DATS=' SITE DAMAGE BY PRE-EMPTIVE ARMS = '
        WRITE(16,1007) DATS,PMDAMG
        DATS='   SITE DAMAGE BY REACTIVE ARMS = '
        WRITE(16,1007) DATS,RMDAMG
 1007   FORMAT(A36,F7.3)
      ENDIF

      WRITE(17,2012) 56,3
 2012 FORMAT(I2,I2)
      WRITE(17,3012) INITDAMG
 3012 FORMAT(F7.4)

      WRITE(17,2013) 57,3
 2013 FORMAT(I2,I2)
      WRITE(17,3013) INITDAMG+TDAMG
 3013 FORMAT(F7.4)

      WRITE(17,2004) 38,3
 2004 FORMAT(I2,I2)
      WRITE(17,3004) TDAMG
 3004 FORMAT(F7.4)

      WRITE(17,2005) 39,3
 2005 FORMAT(I2,I2)
      WRITE(17,3005) ADAMG
 3005 FORMAT(F7.4)

      WRITE(17,2006) 40,3
 2006 FORMAT(I2,I2)
      WRITE(17,3006) SDAMG
 3006 FORMAT(F7.4)

      WRITE(17,2007) 41,3
 2007 FORMAT(I2,I2)
      WRITE(17,3007) DMDAMG
 3007 FORMAT(F7.4)

      WRITE(17,2008) 42,3
 2008 FORMAT(I2,I2)
      WRITE(17,3008) PMDAMG
```

```
3008 FORMAT(F7.4)

     WRITE(17,2009) 43,3
2009 FORMAT(I2,I2)
     WRITE(17,3009) RMDAMG
3009 FORMAT(F7.4)


C.............PRINT OUT NON-SAM GROUND TARGETS FINAL PS.....
C.............IF THIS TYPE OF TARGET IS USED IN THE RUN.....
C.....(i.e. NTARGETS > 0).....

     IF (NTARGETS .EQ. 0) GOTO 10

     TGTDAMG=0.0

     DO 60 L=1,NTARGETS
     TGTDAMG=TGTDAMG+1.0-PST(L)
     IF (DEBUG .GE. 1) THEN
       WRITE(16,1010) L,PST(L)
1010   FORMAT(' NON-SAM GROUND TARGET ',I2,' HAS FINAL PS = ',F6.4)
     ENDIF

     WRITE(17,2010) 47,1
2010 FORMAT(I2,I2)
     WRITE(17,3010) 0.0,0,L,0,0,PST(L)
3010 FORMAT(F5.0,I2,I2,I3,I3,F9.4)

  60 CONTINUE

     IF (DEBUG .GE. 1) THEN
       WRITE(16,1011) TGTDAMG
1011   FORMAT(' TOTAL NON-SAM TARGET DAMAGE  = ',F6.4)
     ENDIF

     WRITE(17,2011) 48,3
2011 FORMAT(I2,I2)
     WRITE(17,3011) TGTDAMG
3011 FORMAT(F7.4)

C ... CALL MESSAGE POINTER SUMMARY SUBROUTINE .....
C     CALL MESSUM

  10 CONTINUE

     IF (IOP(9) .NE. 1) GO TO 99
     CALL PRINT8
     CLOSE(18)
```

```
  99 CONTINUE

C     IF (DEBUG .GE. 1) CLOSE(6)
      IF (DEBUG .GE. 1) CLOSE(16)

C     END FILE 7
C     CLOSE(7)
      END FILE 17
      CLOSE(17)

      END
--------------------------------
      SUBROUTINE TICK2(SDAMG)

C.....LATEST CHANGE MADE  6/20/89 BFS.....

      PARAMETER (IM=30,JM=25,KM=20)

      IMPLICIT LOGICAL (A-Z)

C.....DECLARE LOCAL VARIABLES.....

      REAL*8 RD1
      REAL DI,DP,ELEV,HEAD,PKMSL,VA
      INTEGER CACDED,JJ,L,M,MULTI,N,NC

C.....DECLARE PASSED VARIABLES.....

      REAL SDAMG

C.....DECLARE DOUBLE PRECISION FUNCTION.....

      REAL*8 DRAND

C.....DECLARE VARIABLES IN COMMON BLOCK.....

      REAL BOMPK,BOMTIM,C2DEDTIM,C2ETIM,C2STIM,C2TIME,DT,GDT,LNCH1TIM,
     $LNCH2TIM,PENTARPK,PS,PSA,PSAC,PSAL,PSAMIN,PSSMIN,PST,RRAD,T,
     $TIME,X,XS,XT,Y,YS,YT,Z,ZT

      INTEGER BOM,BOMTGT,CNTRL2,C2END,C2START,C2WPN,DEBUG,I,IOP,ITAR,
     $J,LEG,LNCH1,LNCH2,L1CKPT,L2CKPT,L1WPN,L2WPN,MDEST,MSITE,NAC,
     $NCKPT,NEW,NSITE,PENTYPE,STATS,WHO

C     REAL*8 ISEED
      REAL ISEED
C=====  ABOVE MODIFICATION FOR MAINFRAME VERSION ONLY
==============

C.....COMMON BLOCK.....
```

```
      COMMON/CRUISE/IOP(10),TIMEL,RRAD(JM),IT(JM)
      COMMON/DBLP/ISEED
      COMMON/DEBUG/DEBUG
      COMMON/PLAN/NAC,NCKPT(IM),T(IM,KM),X(IM,KM),Y(IM,KM),Z(IM,KM),
     $V(IM,KM),PSA(IM),PSAMIN,PKM(9),PENTARPK(2,2,20),PSSMIN,
     $MDEST(IM,2),PENTYPE(IM)
      COMMON/SITE/NSITE,XS(JM),YS(JM),ZS(JM),VM(JM),GDT(JM),
     $MSITE(JM),THETJ(JM),CKON1(JM),GTFUS(JM),LMAX(JM),PS(JM),PST(20)
      COMMON/SVSP/ISECT(IM,JM),JAM(IM,JM),ISHOT(IM,JM),DT,
     $XT(IM,2),YT(IM,2),ZT(IM,2),TIME,CD1(IM,JM),CD2(IM,JM),SUST(JM),
     $CD3(IM,JM),CD4(IM,JM),LEG(IM),NEW(IM),I,J,STATS(JM),ITAR(JM)
      COMMON/WEAPONS/BOM(IM),BOMPT(IM,4),BOMTGT(IM,2,4),BOMTIM(IM,4),
     $BOMPK(2,20),NTARGETS,PSAL(IM,2,4),LNCH1(IM),LNCH2(IM),
     $L1CKPT(IM,4),L2CKPT(IM,4),L1WPN(IM,4),L2WPN(IM,4),LNCH1TIM(IM,4),
     $CNTRL2(IM),C2ETIM(IM,4),C2START(IM,4),C2END(IM,4),C2WPN(IM,4),
     $C2TIME(IM,4),LNCH2TIM(IM,4),C2STIM(IM,4),PSAC(IM,4),WHO(IM,2,2,4),
     $C2DEDTIM(IM)

C.....DESCRIPTION OF VARIABLES IN THIS SUBROUTINE.....
C.....                                    .....
C.....PASSED VARIABLES.....
C     SDAMG - TOTAL SAM SITE DAMAGE BY AIR DELIVERED WEAPONS
C.....VARIABLES SET BY PARAMETER STATEMENT.....
C     IM - MAX # OF PENETRATORS
C     JM - MAX # OF SAM SITES
C     KM - MAX # OF PEN. CHECKPOINTS
C.....VARIABLES IN COMMON BLOCK USED IN THIS SUBROUTINE .....
C     BOM(IM) - MAX # OF BOMB GROUPS THAT A/C IM DROPS
C     BOMTGT(IM,2,M) - TARGET TYPE AND LOCATION # FOR BOMB GROUP M
C     BOMTIM(IM,N) - TIME (IN SECONDS) WHEN A/C IM DROPS BOMB GROUP N
C     CNTRL2(IM) - MAX # OF TYPE 2 STANDOFF WEAPONS THAT A/C IM CONTROLS
C     C2DEDTIM(IM) - TIME (IN SECONDS) WHEN CONTROL A/C IM WAS KILLED
C     C2END(IM,N) - CHECKPOINT # WHERE A/C IM ENDS CONTROL OF ITS
C                 - TYPE 2 STANDOFF WEAPON # N
C     C2ETIM(IM,N) - END TIME (IN SECONDS) WHEN AIRCRAFT IM ENDS CONTROL
C                  - OF ITS TYPE 2 STANDOFF WEAPON # N
C     C2START(IM,N) - CHECKPOINT # WHERE A/C IM STARTS CONTROL OF ITS
C                   - TYPE 2 STANDOFF WEAPON # N
C     C2STIM(IM,N) - START TIME (IN SECONDS) WHEN AIRCRAFT IM STARTS
C                  - CONTROL OF A TYPE 2 STANDOFF WEAPON # N
C     C2TIME(IM,N) - TIME (IN SECONDS) BEFORE IMPACT OF CONTROL A/C IM
C                  - WHEN CONTROL OF ITS TYPE 2 STANDOFF WEAPON # N MAY
C                  - BE LOST AND THE WEAPON WILL STILL HIT THE TARGET
C     C2WPN(IM,N) - PENETRATOR # FOR TYPE 2 STANDOFF WEAPON #N THAT A/C
C                 - # IM WILL CONTROL
C     IOP(10) - HOLDS FLAGS(1/0) TO SELECT PROGRAM RUNNING MODES
C     LEG(IM) - HOLDS THE FLIGHTPATH CHECKPOINT DATA LEG NUMBER
C     LNCH1TIM(IM,N) - TIME (IN SECONDS) WHEN A/C IM LAUNCHES ITS TYPE 1
C                    - STANDOFF WEAPONS # N
C     LNCH2TIM(IM,N) - TIME (IN SECONDS) WHEN A/C IM LAUNCHES ITS TYPE 2
```

```
C                - STANDOFF WEAPONS # N
C     LNCH1(IM) - MAX # OF TYPE 1 STANDOFF WEAPONS A/C IM LAUNCHES
C     LNCH2(IM) - MAX # OF TYPE 2 STANDOFF WEAPONS A/C IM LAUNCHES
C     L1CKPT(IM,N) - CHECKPOINT # WHERE A/C IM LAUNCHES ITS TYPE 1
C              - STANDOFF WEAPON # N
C     L2CKPT(IM,N) - CHECKPOINT # WHERE A/C IM LAUNCHES ITS TYPE 2
C              - STANDOFF WEAPON # N
C     L1WPN(IM,N) - PENETRATOR # FOR TYPE 1 STANDOFF WEAPON # N THAT A/C
C              - # IM WILL LAUNCH
C     L2WPN(IM,N) - PENETRATOR # FOR TYPE 2 STANDOFF WEAPON # N THAT A/C
C              - # IM WILL LAUNCH
C     MDEST(IM,2) - FOR PENETRATOR IM;
C              - IF MDEST(IM,1)=1; PENETRATOR IS TARGETTED AGAINST
C              - A SAM SITE AND MDEST(IM,2) HOLDS THE SITE #.
C              - IF MDEST(IM,1)=2; PENETRATOR IS TARGETTED AGAINST
C              - A NON-SAM SITE TYEP TARGET AND MDEST(IM,2) HOLDS
C              - THE TARGET #.
C     MSITE(JM) - HOLDS THE SAM SITE TYPE (1-9) FOR SITE JM
C     NAC - MAX # OF PENETRATORS
C     NCKPT(IM)
C     NEW(IM)
C     NSITE - MAX # OF PENETRATORS
C     PENTARPK(2,2,20) - PK OF EACH PENETRATOR TYPE FOR EACH SAM SITE
C                  - TYPE OR EACH NON-AD TARGET NUMBER
C     PENTYPE(IM) - INTEGER INDICATING THE PENETRATOR TYPE
C     PSAC(IM,N) - PROBABILITY OF SURVIVAL OF THE CONTROL AIRCRAFT IM AT
C              - THE TIME THIS AIRCRAFT REACHES THE CONTROL POINT FOR
C              - STANDOFF WEAPON N
C     PSAL(IM,L,N) - PROBABILITY OF SURVIVAL OF THE LAUNCH AIRCRAFT IM
C               - AT THE TIME A TYPE L STANDOFF WEAPON N IS LAUNCHED
C     PSAMIN - MINIMUM SURVIVABILITY OF PENETRATOR;IF PSA(IM) IS BELOW
C          - THIS, THE PENETRATOR IS REMOVED FROM THE SIMULATION
C     PSA(IM) - SURVIVAL PROBABILITY OF PENETRATOR IM
C     PS(JM) - SURVIVAL PROBABILITY OF SAM SITE JM
C     PSSMIN - MIN SURVIVAL PROBABILITY THAT THE SAM SITE MUST MAINTAIN
C          - IN ORDER TO STAY IN THE SIMULATION
C     PST(20) - PS OF NON-SAM TYPE TARGET
C     RRAD(JM) - HOLDS TIMING DATA (IN SECONDS) FOR SAM SITE JM; USED
C           - IN DETERMINING WHEN TO FIRE,RE-FIRE,TRACK,ETC...
C     STATS(JM) - INTEGER VALUE INDICATING THE STATUS OF EACH SAM SITE
C              - IF =0; SITE IS NOT RADIATING
C              - IF =1; SITE IS ESTABLING A TRACK FILE ON A TARGET
C              - AND GETTING READY TO FIRE
C              - IF =2; SITE HAS A SALVO ENROUTE TO THE TARGET
C              - IF =3; SITE IS PERFORMING POST INTERCEPT DECISIONS
C     T(IM,KM)
C     TIME - SIMULATION TIME (IN SECONDS) FROM THE START OF THE RUN
C     WHO(IM,J,K,N) - HOLDS DATA ON WHO LAUNCHES AND CONTROLS WHOM.
C                - FOR A/C IM: INPUT ARE:
C                -  J-IS STANDOFF WEAPON TYPE (1 OR 2)
```

bilities that K-way and higher effects are zero

DF	L. R. Chisq	Prob	Pearson Chisq	Prob	Iteration
10	5.9295	.8212	5.9811	.8168	8
47	39.9497	.7572	37.8685	.8266	5
99	125.2471	.0385	119.3168	.0804	5
133	254.0287	.0000	244.2417	.0000	2
143	1161.1507	.0000	1090.9009	.0000	2

bilities that all K-way effects are zero

DF	L. R. Chisq	Prob	Pearson Chisq	Prob
10	5.9295	.8212	5.9811	.8168
37	34.0202	.6095	31.8874	.7073
52	85.2974	.0025	81.4483	.0056
34	128.7816	.0000	124.9249	.0000
10	907.1220	.0000	846.6591	.0000

```
C               -  WHERE K = 1; DATA IS THE WEAPON # IT LAUNCHES
C               -  WHERE K = 2; DATA IS THE WEAPON # IT CONTROLS
C               -  N IS THE WEAPON #(1-4) ASSIGNED TO THAT A/C
C               -  OUTPUT IS : STANDOFF WEAPON PEN. #
C              - FOR STANDOFF WEAPON IM: INPUTS ARE
C               -  J-IS STANDOFF WEAPON TYPE (2 IS NORMALLY USED)
C               -  WHERE K = 1; DATA IS THE LAUNCH A/C #
C               -  WHERE K = 2; DATA IS THE CONTROL A/C #
C               -  FOR N=1; OUTPUT IS A/C # IF INTEREST
C               -  FOR N=2; OUTPUT IS WEAPON # OF A/C OF INTEREST
C     X(IM,KM)
C     XS(JM) - X POSITION (IN NMI) OF SAM SITE JM
C     XT(IM,2)
C     Y(IM,KM)
C     YS(JM) - Y POSITION (IN NMI) OF SAM SITE JM
C     YT(IM,2)
C     Z(IM,KM)
C     ZT(IM,2)
C.....VARIABLES LOCAL TO THIS SUBROUTINE.....
C     CACDED - FLAG (1/0) TO INDICATE THAT THE TYPE 2 WEAPON IMPACTED
C          - THE GROUND AND WAS INEFFECTIVE AGAINST THE TARGET BECAUSE
C          - ITS CONTROL AIRCRAFT HAD BEEN DESTROYED WELL BEFORE
C          - WEAPON IMPACT
C     DI
C     DP
C     ELEV
C     HEAD
C     JJ
C     L
C     M
C     MULTI - 1/0 FLAG IF SITE IS PART OF A MULTITARGETTING GROUP
C     N
C     NC
C     RD1
C     PKMSL - HOLDS PK VALUE FROM PENTARPK ARRAY
C     VA - FRACTION OF TIME LEFT BETWEEN PREVIOUS CHECKPOINT AND THE
C       - NEXT CHECKPOINT MEASURED FROM THE NEXT CHECKPOINT
C.....START PROGRAM.....

C     WRITE(*,*) 'TICK2'

C.....UPDATE CLOCK.....

      TIME=TIME+DT

      DO 10 I=1,NAC

      IF (CNTRL2(I) .GE. 1) THEN
        DO 100 M=1,CNTRL2(I)
        IF (C2ETIM(I,M) .GE. TIME .AND. C2ETIM(I,M) .LT. TIME+DT) THEN
```

```
          PSAC(I,M)=PSA(I)
          IF (PSA(I) .GE. PSAMIN) THEN
            IF (DEBUG .GE. 1) THEN
              WRITE(16,1010) TIME,I,WHO(I,2,2,M),PSAC(I,M)
1010            FORMAT(F5.0,' SEC A/C  ',I2,' ENDS CONTROL FUNCTION ',
     $          'OF PEN. # ',I2,' ; A/C PS = ',F4.2)
            ENDIF
            WRITE(17,2010) 62,1
2010          FORMAT(I2,I2)
            WRITE(17,3010) TIME,I,0,WHO(I,2,2,M),0,PSAC(I,M)
3010          FORMAT(F5.0,I2,I2,I3,I3,F9.4)
          ENDIF
        ENDIF
 100    CONTINUE
      ENDIF

C..... IF THE A/C HAS BEEN KILLED, WAS IT SUPPOSED TO DO SOMETHING?..

      IF (PSA(I) .LT. PSAMIN) THEN

        IF (BOM(I) .GE. 1) THEN
          DO 110 M=1,BOM(I)
          IF (TIME .LT. BOMTIM(I,M)) THEN
            BOM(I)=0
            IF (DEBUG .GE. 1) THEN
              WRITE(16,1017) TIME,I
1017            FORMAT(F5.0,' SEC A/C  ',I2,' IS KILLED AND WILL NOT',
     $          ' DROP ITS BOMBS')
            ENDIF
            WRITE(17,2017) 68,1
2017          FORMAT(I2,I2)
            WRITE(17,3017) TIME,I,0,0,0,0.0
3017          FORMAT(F5.0,I2,I2,I3,I3,F9.4)
          ENDIF
 110      CONTINUE
        ENDIF

        IF (LNCH1(I) .GE. 1) THEN
          DO 120 M=1,LNCH1(I)
          IF (TIME .LT. LNCH1TIM(I,M)  .AND.
     $      PSA(L1WPN(I,M)) .NE. 0.0) THEN
            PSA(L1WPN(I,M))=0.0
            IF (DEBUG .GE. 1) THEN
              WRITE(16,1016) TIME,I,L1WPN(I,M)
1016            FORMAT(F5.0,' SEC A/C  ',I2,' IS KILLED AND WILL NOT',
     $          ' LAUNCH PEN. # ',I2,' (TYPE 1 WEAPON)')
            ENDIF
            WRITE(17,2016) 67,1
2016          FORMAT(I2,I2)
            WRITE(17,3016) TIME,I,0,L1WPN(I,M),0,0.0
```

```
3016        FORMAT(F5.0,I2,I2,I3,I3,F9.4)
          ENDIF
 120       CONTINUE
        ENDIF


        IF (LNCH2(I) .GE. 1) THEN
          DO 130 M=1,LNCH2(I)
          IF (TIME .LT. LNCH2TIM(I,M) .AND.
     $       PSA(L2WPN(I,M)) .NE. 0.0) THEN
            PSA(L2WPN(I,M))=0.0
            IF (DEBUG .GE. 1) THEN
              WRITE(16,1011) TIME,I,L2WPN(I,M)
1011            FORMAT(F5.0,' SEC A/C  ',I2,' IS KILLED AND WILL NOT',
     $          ' LAUNCH PEN. # ',I2,' (TYPE 2 WEAPON)')
            ENDIF
            WRITE(17,2011) 63,1
2011         FORMAT(I2,I2)
            WRITE(17,3011) TIME,I,0,L2WPN(I,M),0,0.0
3011         FORMAT(F5.0,I2,I2,I3,I3,F9.4)
          ENDIF
 130       CONTINUE
        ENDIF

        IF (CNTRL2(I) .GE. 1) THEN
          DO 140 M=1,CNTRL2(I)
          IF (TIME .LT. C2ETIM(I,M) .AND.
     $       C2DEDTIM(I) .EQ. 0) THEN

            IF (DEBUG .GE. 1) THEN
              WRITE(16,1012) TIME,I,C2WPN(I,M)
1012            FORMAT(F5.0,' SEC A/C  ',I2,' IS KILLED AND WILL NOT',
     $          ' CONTROL PEN. # ',I2,' (TYPE 2 WEAPON)')
            ENDIF
            WRITE(17,2012) 64,1
2012         FORMAT(I2,I2)
            WRITE(17,3012) TIME,I,0,C2WPN(I,M),0,0.0
3012         FORMAT(F5.0,I2,I2,I3,I3,F9.4)

C       IF IOP(4) = 1 AND IF THE CONTROL A/C HAS BEEN DESTROYED BEFORE
C       NOW, THE LAUNCH A/C DOES NOT LAUNCH A WEAPON. OTHERWISE THE
C       WEAPON IS LAUNCHED BUT IS NOT CONTROLLED AND HAS NO ATTRIATION
C       AGAINST THE TARGET ALTHOUGH THE SAMS MAY ENGAGE IT.
C
            IF (IOP(4) .EQ. 1) THEN
              PSA(C2WPN(I,M))=0.0
              LNCH2TIM(WHO(C2WPN(I,M),2,1,1),WHO(C2WPN(I,M),2,1,2))=0.0
              IF (DEBUG .GE. 1) THEN
                WRITE(16,1018) C2WPN(I,M)
1018             FORMAT(' THE CONTROL A/C IS KILLED SO PEN. # ',I2,
```

```
     $          ' WILL NOT BE LAUNCHED')
               ENDIF
               WRITE(17,2018) 79,1
2018            FORMAT(I2,I2)
               WRITE(17,3018) 0.0,C2WPN(I,M),0,0,0,0.0
3018            FORMAT(F5.0,I2,I2,I3,I3,F9.4)
              ENDIF

            ENDIF
 140         CONTINUE
            DO 145 M=1,CNTRL2(I)
            IF (TIME .LT. C2ETIM(I,M) .AND. C2DEDTIM(I) .EQ. 0)
     $      C2DEDTIM(I)=TIME
 145         CONTINUE
          ENDIF

          GO TO 10

        ENDIF

C.....TIME TO DROP BOMBS ?.....

        IF (BOM(I) .GE. 1) THEN
          DO 150 M=1,BOM(I)
          IF (BOMTIM(I,M) .GE. TIME  .AND.
     $      BOMTIM(I,M) .LT. TIME+DT) THEN
            IF (BOMTGT(I,1,M) .EQ. 1) THEN
              J=BOMTGT(I,2,M)
C              RD1=DRAND(ISEED)
              RD1=ISEED
C= = = = = = DRAND USED IN PC VERSION NOT MAINFRAME VERSION = = = = = = = = = =
              DP=PS(J)
              IF (IOP(1) .EQ. 0)  PS(J)=PS(J)*(1.0-PSA(I)*
     $        BOMPK(1,MSITE(J)))
              IF (IOP(1) .NE. 0  .AND.  RD1 .LT. PSA(I)*
     $        BOMPK(1,MSITE(J)))
     $        PS(J)=0.0
              DP=DP-PS(J)
              SDAMG=SDAMG+DP
              IF (DEBUG .GE. 1) THEN
                WRITE (16,1005) TIME,I,J,PS(J),RD1,XS(J),YS(J)
1005              FORMAT(F5.0,' SEC A/C  ',I2,' BOMBS SAM SITE ',I2,
     $            ' PS = ',F4.2,' RND#',F4.2,2F6.1)
              ENDIF
              WRITE(17,2005) 49,1
2005            FORMAT(I2,I2)
              WRITE(17,3005) TIME,I,J,0,0,PS(J)
3005            FORMAT(F5.0,I2,I2,I3,I3,F9.4)
            ELSEIF (BOMTGT(I,1,M) .EQ. 2) THEN
              L=BOMTGT(I,2,M)
```

```
C            RD1=DRAND(ISEED)
            RD1=ISEED
C= = = = = = DRAND USED IN PC VERSION NOT MAINFRAME VERSION = = = = = = = = = =
            IF (IOP(1) .EQ. 0) PST(L)=PST(L)*(1.0-PSA(I)*BOMPK(2,L))
            IF (IOP(1) .NE. 0  .AND.  RD1 .LT. PSA(I)*BOMPK(2,L))
     $       PST(L)=0.0
            IF (DEBUG .GE. 1) THEN
              WRITE (16,1006) TIME,I,L,PST(L),RD1,XS(J),YS(J)
1006           FORMAT(F5.0,' SEC A/C  ',I2,' BOMBS NON-SAM TARGET  ',
     $         I2,' PS = ',F4.2,' RND#',F4.2,2F6.1)
            ENDIF
            WRITE(17,2006) 50,1
2006         FORMAT(I2,I2)
            WRITE(17,3006) TIME,I,L,0,0,PST(L)
3006         FORMAT(F5.0,I2,I2,I3,I3,F9.4)
          ENDIF
        ENDIF
 150    CONTINUE
      ENDIF


C.....TIME TO LAUNCH STANDOFF WEAPON TYPE 1 ?.....


      IF (LNCH1(I) .GE. 1) THEN
        DO 160 M=1,LNCH1(I)
        IF (LNCH1TIM(I,M) .GE. TIME  .AND.
     $     LNCH1TIM(I,M) .LT. TIME+DT) THEN
          PSAL(I,1,M)=PSA(I)
          IF (DEBUG .GE. 1) THEN
            WRITE (16,1007) TIME,I,L1WPN(I,M),PSA(I)
1007         FORMAT(F5.0,' SEC A/C  ',I2,' LAUNCHES PEN. # ',I2,
     $       ' (TYPE 1 WEAPON)',' ;A/C PS =',F4.2)
          ENDIF
          WRITE(17,2007) 51,1
2007       FORMAT(I2,I2)
          WRITE(17,3007) TIME,I,0,L1WPN(I,M),0,PSA(I)
3007       FORMAT(F5.0,I2,I2,I3,I3,F9.4)
        ENDIF
 160    CONTINUE
      ENDIF

C.....TIME TO LAUNCH STANDOFF WEAPON TYPE 2 ?.....

      IF (LNCH2(I) .GE. 1) THEN
        DO 170 M=1,LNCH2(I)
        IF (LNCH2TIM(I,M) .GE. TIME  .AND.
     $     LNCH2TIM(I,M) .LT. TIME+DT) THEN
          PSAL(I,2,M)=PSA(I)
          IF (DEBUG .GE. 1) THEN
```

```
            WRITE (16,1008) TIME,I,L2WPN(I,M),PSA(I)
1008          FORMAT(F5.0,' SEC A/C  ',I2,' LAUNCHES PEN. # ',I2,
     $       ' (TYPE 2 WEAPON)',';A/C PS =',F4.2)
          ENDIF
          WRITE(17,2008) 60,1
2008       FORMAT(I2,I2)
          WRITE(17,3008) TIME,I,0,L2WPN(I,M),0,PSA(I)
3008       FORMAT(F5.0,I2,I2,I3,I3,F9.4)
        ENDIF
 170    CONTINUE
      ENDIF


C.....TIME TO START CONTROL STANDOFF WEAPON TYPE 2 ?.....

      IF (CNTRL2(I) .GE. 1) THEN
        DO 180 M=1,CNTRL2(I)
        IF (C2STIM(I,M) .GE. TIME  .AND.
     $     C2STIM(I,M) .LT. TIME+DT) THEN
          PSAC(I,M)=PSA(I)
          IF (DEBUG .GE. 1) THEN
            WRITE (16,1009) TIME,I,C2WPN(I,M),PSAC(I,M)
1009          FORMAT(F5.0,' SEC A/C  ',I2,' STARTS CONTROLING PEN. # ',
     $        I2,' (TYPE 2 WEAPON)',' ;A/C PS =',F4.2)
          ENDIF
          WRITE(17,2009) 61,1
2009       FORMAT(I2,I2)
          WRITE(17,3009) TIME,I,0,C2WPN(I,M),0,PSAC(I,M)
3009       FORMAT(F5.0,I2,I2,I3,I3,F9.4)
        ENDIF
 180    CONTINUE
      ENDIF

C.....UPDATE AIRCRAFT POSITION.....

      LEG(I)=0
      IF (TIME .LE. T(I,1)) GO TO 10
      LEG(I)=1
      XT(I,1)=XT(I,2)
      YT(I,1)=YT(I,2)
      ZT(I,1)=ZT(I,2)
      NC=NCKPT(I)


      DO 20 N=2,NC
      IF (TIME .LE. T(I,N)) GO TO 30
      LEG(I)=LEG(I)+1
   20 CONTINUE

C.....
```

```
      IF(MDEST(I,1) .EQ. 0  .OR.  N .LT. NC) GO TO 10

C.....STANDOFF WEAPON IS NOW AT THE END OF ITS FLIGHTPATH.....
C.....DETERMINE EFFECTS OF IMPACT OF STANDOFF WEAPON .....
C.....AGAINST SAM SITES .....

      CACDED=0

      IF(MDEST(I,1) .EQ. 1) THEN
        J=MDEST(I,2)
        DP=PS(J)
C             RD1=DRAND(ISEED)
              RD1=ISEED
C= = = = = = DRAND USED IN PC VERSION NOT MAINFRAME VERSION = = = = = = = = = =
        PKMSL=PENTARPK(PENTYPE(I),1,MSITE(J))
        IF (PENTYPE(I) .EQ. 1) THEN
          IF (IOP(1) .EQ. 0) PS(J)=PS(J)*(1.0-PKMSL*PSA(I)*
     $     PSAL(WHO(I,1,1,1),1,WHO(I,1,1,2)))
        ELSEIF (PENTYPE(I) .EQ. 2) THEN
          IF (PSA(WHO(I,2,2,1)) .GE. PSAMIN) THEN
            IF (TIME .LE. C2ETIM(WHO(I,2,2,1),WHO(I,2,2,2))) THEN
              IF (IOP(1) .EQ. 0) PS(J)=PS(J)*(1.0-PKMSL*PSA(I)*
     $           PSAL(WHO(I,2,1,1),2,WHO(I,2,1,2))*(PSAC(
     $           WHO(I,2,2,1),WHO(I,2,2,2))+PSA(WHO(I,2,2,1)))/2.0)
            ELSEIF (TIME .GT. C2ETIM(WHO(I,2,2,1),WHO(I,2,2,2))) THEN
              IF (IOP(1) .EQ. 0) PS(J)=PS(J)*(1.0-PKMSL*PSA(I)*
     $           PSAL(WHO(I,2,1,1),2,WHO(I,2,1,2))*
     $           PSAC(WHO(I,2,2,1),WHO(I,2,2,2)))
            ENDIF
          ELSEIF (PSA(WHO(I,2,2,1)) .LT. PSAMIN) THEN
            IF (TIME-C2DEDTIM(WHO(I,2,2,1)) .LT.
     $           C2TIME(I,WHO(I,2,2,2))) THEN
              IF (IOP(1) .EQ. 0) PS(J)=PS(J)*(1.0-PKMSL*PSA(I)*
     $           PSAL(WHO(I,2,1,1),2,WHO(I,2,1,2))*PSAC(
     $           WHO(I,2,2,1),WHO(I,2,2,2))*(C2TIME(I,WHO(I,2,2,2))-
     $           (TIME-C2DEDTIM(WHO(I,2,2,1))))/C2TIME(I,WHO(I,2,2,2)))
            ELSEIF (TIME-C2DEDTIM(WHO(I,2,2,1)) .GE.
     $           C2TIME(I,WHO(I,2,2,2))) THEN
              IF (DEBUG .GE. 1) THEN
                WRITE(16,1013) TIME,I,WHO(I,2,2,1)
1013              FORMAT(F5.0,' SEC PEN. ',I2,' (TYPE 2',
     $            ' WPN) IMPACTS GROUND; CONTROL A/C ',I2,
     $            ' KILLED PREVIOUSLY')
              ENDIF
              CACDED=1
              WRITE(17,2013) 65,1
2013            FORMAT(I2,I2)
              WRITE(17,3013) TIME,I,0,WHO(I,2,2,1),0,0.0
3013            FORMAT(F5.0,I2,I2,I3,I3,F9.4)
            ENDIF
```

```
          ENDIF
        ENDIF
        IF (IOP(1) .NE. 0  .AND.  RD1 .LT. PKMSL) PS(J)=0.0
        DP=DP-PS(J)
        SDAMG=SDAMG+DP
        PSA(I)=0.0

C.....***...CHECK SITE IF IT PART OF A MULTITARGETTING GROUP.....

      CALL MULSAM(MULTI)

C.....***.....

C.....OR FOR NON-SAM TARGETS.....

      ELSEIF (MDEST(I,1) .EQ. 2) THEN
        L=MDEST(I,2)
C             RD1=DRAND(ISEED)
              RD1=ISEED
C= = = = = = DRAND USED IN PC VERSION NOT MAINFRAME VERSION = = = = = = = = = =
        PKMSL=PENTARPK(PENTYPE(I),2,L)
        IF (PENTYPE(I) .EQ. 1) THEN
          IF (IOP(1) .EQ. 0) PST(L)=PST(L)*(1.0-PKMSL*PSA(I)*
     $     PSAL(WHO(I,1,1,1),1,WHO(I,1,1,2)))
        ELSEIF (PENTYPE(I) .EQ. 2) THEN
          IF (PSA(WHO(I,2,2,1)) .GE. PSAMIN) THEN
            IF (TIME .LE. C2ETIM(WHO(I,2,2,1),WHO(I,2,2,2))) THEN
              IF (IOP(1) .EQ. 0) PST(L)=PST(L)*(1.0-PKMSL*PSA(I)*
     $          PSAL(WHO(I,2,1,1),2,WHO(I,2,1,2))*(PSAC(
     $          WHO(I,2,2,1),WHO(I,2,2,2))+PSA(WHO(I,2,2,1)))/2.0)
            ELSEIF (TIME .GT. C2ETIM(WHO(I,2,2,1),WHO(I,2,2,2))) THEN
              IF (IOP(1) .EQ. 0) PST(L)=PST(L)*(1.0-PKMSL*PSA(I)*
     $          PSAL(WHO(I,2,1,1),2,WHO(I,2,1,2))*
     $          PSAC(WHO(I,2,2,1),WHO(I,2,2,2)))
            ENDIF
          ELSEIF (PSA(WHO(I,2,2,1)) .LT. PSAMIN) THEN
            IF (TIME-C2DEDTIM(WHO(I,2,2,1)) .LT.
     $          C2TIME(I,WHO(I,2,2,2))) THEN
              IF (IOP(1) .EQ. 0) PST(L)=PST(L)*(1.0-PKMSL*PSA(I)*
     $          PSAL(WHO(I,2,1,1),2,WHO(I,2,1,2))*PSAC(
     $          WHO(I,2,2,1),WHO(I,2,2,2))*(C2TIME(I,WHO(I,2,2,2))-
     $          (TIME-C2DEDTIM(WHO(I,2,2,1))))/C2TIME(I,WHO(I,2,2,2)))
            ELSEIF (TIME-C2DEDTIM(WHO(I,2,2,1)) .GE.
     $          C2TIME(I,WHO(I,2,2,2))) THEN
              IF (DEBUG .GE. 1) THEN
                WRITE(16,1014) TIME,I,WHO(I,2,2,1)
 1014             FORMAT(F5.0,' SEC PEN. ',I2,' (TYPE 2',
     $            ' WPN) IMPACTS GROUND; CONTROL A/C ',I2,
     $            ' KILLED PREVIOUSLY')
              ENDIF
```

```
              CACDED=1
              WRITE(17,2014) 65,1
2014           FORMAT(I2,I2)
              WRITE(17,3014) TIME,I,0,WHO(I,2,2,1),0,0.0
3014           FORMAT(F5.0,I2,I2,I3,I3,F9.4)
            ENDIF
          ENDIF
        ENDIF
        IF (IOP(1) .NE. 0  .AND.  RD1 .LT. PKMSL) PST(L)=0.0
        PSA(I)=0.0
      ENDIF

      DO 40 JJ=1,NSITE

      IF(ITAR(JJ) .NE. I) GO TO 40
      STATS(JJ)=3
      RRAD(JJ)=GDT(JJ)+1.0

C.....SITE LOSES PEN. AS A TARGET WHEN THE PEN. HITS ITS INTENDED
C..... GROUND TARGET AND ADDED 'PS(J) .GT. PSSMIN' 9/18/87 BFS..

      IF (DEBUG .GE. 1  .AND.  PS(J) .GT. PSSMIN) THEN
        WRITE (16,1003) TIME,JJ,I
1003    FORMAT(F5.0,' SEC SITE ',I2,' LOSES PEN. ',I2,
     $  ' :PEN. IMPACTS GROUND TARGET')
      ENDIF

      WRITE(17,2003) 46,1
2003 FORMAT(I2,I2)
      WRITE(17,3003) TIME,I,JJ,0,0,0.0
3003 FORMAT(F5.0,I2,I2,I3,I3,F9.4)

   40 CONTINUE


C.....PRINTOUT RESULTS OF WEAPON IMPACT.....

      IF (CACDED .EQ. 1) GO TO 10

C..***  RESULTS ALREADY PRINTED OUT IF SITE IS PART OF MULTITARGET GROUP

      IF (MULTI .EQ. 1) GO TO 10

C.....***.....

      IF (MDEST(I,1) .EQ. 1) THEN
        IF (PS(J) .LT. PSSMIN) THEN
          IF (DEBUG .GE. 1) THEN
            WRITE (16,1000) TIME,I,J,PS(J),RD1,XS(J),YS(J)
1000          FORMAT(F5.0,' SEC STANDOFF WEAPON ',I2,' DESTROYS SITE ',
```

```
     $       I2,' PS=',F4.2,' RND#',F4.2,2F6.1)
           ENDIF
           WRITE(17,2000) 45,1
2000        FORMAT(I2,I2)
           WRITE(17,3000) TIME,I,J,0,0,PS(J)
3000        FORMAT(F5.0,I2,I2,I3,I3,F9.4)
         ELSEIF (PS(J) .GE. PSSMIN) THEN
           IF (DEBUG .GE. 1) THEN
             WRITE (16,1001) TIME,I,J,PS(J),RD1,XS(J),YS(J)
1001          FORMAT(F5.0,' SEC STANDOFF WEAPON ',I2,' IMPACTS SITE ',
     $       I2,' PS=',F4.2,' RND#',F4.2,2F6.1)
           ENDIF
           WRITE(17,2001) 33,1
2001        FORMAT(I2,I2)
           WRITE(17,3001) TIME,I,J,0,0,PS(J)
3001        FORMAT(F5.0,I2,I2,I3,I3,F9.4)
         ENDIF
       ELSEIF (MDEST(I,1) .EQ. 2) THEN
         IF (PST(L) .LT. PSSMIN) THEN
           IF (DEBUG .GE. 1) THEN
             WRITE (16,1015) TIME,I,L,PST(L),RD1
1015          FORMAT(F5.0,' SEC STANDOFF WEAPON ',I2,' DESTROYS ',
     $       'NON-SAM TARGET ',I2,' PS=',F4.2,' RND#',F4.2)
           ENDIF
           WRITE(17,2015) 66,1
2015        FORMAT(I2,I2)
           WRITE(17,3015) TIME,I,L,0,0,PST(L)
3015        FORMAT(F5.0,I2,I2,I3,I3,F9.4)
         ELSEIF (PST(L) .GE. PSSMIN) THEN
           IF (DEBUG .GE. 1) THEN
             WRITE (16,1004) TIME,I,L,PST(L),RD1
1004          FORMAT(F5.0,' SEC STANDOFF WEAPON ',I2,' IMPACTS NON-SAM'
     $       ,' TARGET ',I2,' PS=',F4.2,' RND#',F4.2)
           ENDIF
           WRITE(17,2004) 59,1
2004        FORMAT(I2,I2)
           WRITE(17,3004) TIME,I,L,0,0,PST(L)
3004        FORMAT(F5.0,I2,I2,I3,I3,F9.4)
         ENDIF
       ENDIF

C ... IF USING IADS MODE, UPDATE MESSAGE TIME DUE TO NEW DAMAGE LEVEL..
       IF(IOP(3) .GE. 5) THEN
         CALL IADSPS(MDEST(I,1),J,L,RND1)
       ENDIF

C.....

       GO TO 10
```

```
 30 CONTINUE

    VA=(T(I,N)-TIME)/(T(I,N)-T(I,N-1))
    XT(I,2) = X(I,N) -(X(I,N)-X(I,N-1))*VA
    YT(I,2) = Y(I,N) -(Y(I,N)-Y(I,N-1))*VA
    ZT(I,2) = Z(I,N) -(Z(I,N)-Z(I,N-1))*VA
    NEW(I)=1


C.....NEW(I)=1 MEANS COURSE HAS RECENTLY CHANGED.....

    IF (TIME-1.0*DT .GT. T(I,N-1)) NEW(I)=0
    IF (NEW(I) .EQ. 0) GO TO 10
    IF((XT(I,2)-X(I,N)) .EQ. (YT(I,2)-Y(I,N)) .AND.
   &   (XT(I,2)-X(I,N)) .EQ. 0.0)  THEN
      HEAD=0.0
    ELSE
      HEAD= ATAN2(XT(I,2)-X(I,N),YT(I,2)-Y(I,N))*57.29578
    ENDIF
    IF (HEAD .LT. 0.) HEAD=HEAD+360.
    DI=(XT(I,2)-X(I,N-1))*(XT(I,2)-X(I,N-1))+(YT(I,2)-Y(I,N-1))*
   $(YT(I,2)-Y(I,N-1))
    DI=SQRT(DI)
    IF((ZT(I,2)-Z(I,N-1)) .EQ. DI  .AND.  DI .EQ. 0.0) THEN
      ELEV=0.0
    ELSE
      ELEV=ATAN2(ZT(I,2)-Z(I,N-1),DI)*57.29578
    ENDIF

    IF (DEBUG .GE. 1) THEN
      WRITE (16,1002) TIME,I,HEAD,ELEV,XT(I,2),YT(I,2),ZT(I,2)
1002   FORMAT(F5.0,' SEC PEN. ',I2,' CHANGES COURSE',F6.1,'=AZ ',
   $  F6.1,'=EL ',3F8.2)
    ENDIF
C.....WRITE EVENT AND DATA TO DOC FILE.....

    WRITE(17,2002) 34,1
2002 FORMAT(I2,I2)
    WRITE(17,3002) TIME,I,0,0,0,HEAD
3002 FORMAT(F5.0,I2,I2,I3,I3,F9.4)

    DO 50 J=1,NSITE
    CALL GENRAL
 50 CONTINUE

 10 CONTINUE

    RETURN
    END
-----------------------------------------------
```

```
C
C  PARAMETERS OF AWDDATA DEF
C
C PSADWOC - CURRENT PROBABILITY OF SURVIVAL OF THE ADWOC CENTER
C MINADWOCPS - MINIMUM SURVIVAL PROBABILITY FOR ADWOC TO SAY IN SIM.
C ADWPROCDLY - NORMAL PROCESSING DELAY FROM DETECTION OF PENETRATOR
C          - UNTIL ADWOC CENTER READY TO SEND MESSAGE (SECONDS)
C ADWDAMDLY - MAX. EXTRA DELAY UNTIL ADWOC CENTER READY TO SEND
C         - MESSAGE DUE TO SITE BEING DAMAGED
C ADWDAMRAT - SCALE FACTOR USED WITH ADWDAMDLY(9) TO COMPUTE DELAY
C         - DUE TO SITE DAMAGE
C
      REAL ADWPROCDLY,ADWDAMDLY,ADWDAMRAT,
     & PSADWOC,MINADWOCPS

      COMMON/ADWDATA/ADWPROCDLY,ADWDAMDLY,ADWDAMRAT,
     & PSADWOC,MINADWOCPS
C
----------------------------------
C
C  PARAMETERS OF ADWMES1 DEF
C
C  ADWMESID(MAXADWMESID)-
C   - HOLDS THE ID NUMBER OF EACH UNIQUE MESSAGE TO THE ADWOC CENTER.
C   - USED TO COMPARE INCOMING MESSAGES AND REJECT PREVIOUSLY
C   - RECEIVED MESSAGES
C  MAXADWMESID - MAX SIZE OF RECEIVED ID ARRAY
C ADWMESIDPOS - ARRAY POINTER WHERE NEXT MESSAGE ID# MAY BE INSERTED
C
      INTEGER MAXADWMESID,ADWMESID,ADWMESIDPOS
      PARAMETER(MAXADWMESID=150)

      COMMON/ADWMES1/ADWMESID(MAXADWMESID),ADWMESIDPOS
C
----------------------------------
C
C  PARAMETERS OF BATDATA DEF
C
C NUMBAT - MAX NUMBER OF BATTALION CENTERS IN THIS SIMULATION.
C MINBATPS - MIN. SURVIVAL PROB. FOR BATTALION CENTERS TO SAY IN SIM.
C BATPROCDLY - NORMAL PROCESSING DELAY FROM DETECTION OF PENETRATOR
C          - UNTIL BATTALION CENTER READY TO SEND MESSAGE (SECONDS)
C BATDAMDLY(9) - MAX. EXTRA DELAY UNTIL BATTALION CENTER READY TO SEND
C            - MESSAGE DUE TO BATTALION CENTER BEING DAMAGED
C BATDAMRAT(9) - SCALE FACTOR USED WITH BATDAMDLY(9) TO COMPUTE DELAY
C            - DUE TO BATTALION CENTER DAMAGE
C
      REAL BATPROCDLY,BATDAMDLY,BATDAMRAT,MINBATPS
      INTEGER NUMBAT
```

```
      COMMON/BATDATA/BATPROCDLY(9),BATDAMDLY(9),BATDAMRAT(9),
     & MINBATPS,NUMBAT
C
-------------------------------
C
C   PARAMETERS OF BATMES1 DEF
C
C   BATMESID(9,MAXBATMESID)-
C     - HOLDS THE ID NUMBER OF EACH UNIQUE MESSAGE TO EACH BATTALION
C     - CENTER. USED TO COMPARE INCOMING MESSAGES AND REJECT PREVIOUSLY
C     - RECEIVED MESSAGES
C   MAXBATMESID - MAX SIZE OF RECEIVED ID ARRAY
C  BATMESIDPOS(9) - ARRAY POINTER WHERE NEXT MESSAGE ID# MAY BE INSERTED
C                 - FOR EACH BATTALION CENTER
C
      INTEGER MAXBATMESID,BATMESID,BATMESIDPOS
      PARAMETER(MAXBATMESID=50)

      COMMON/BATMES1/BATMESID(9,MAXBATMESID),BATMESIDPOS(9)
C
-------------------------------
C
C   PARAMETERS OF CON1 DEF
C
C  CONNECT(MAXCON,4) - HOLDS IADS CONNECTION STRUCTURE. INDICATES WHERE
C                - EACH IADS NODE CAN SEND A MESSAGE TO. FIRST PAIR #'S
C                - IDENTIFY SENDER AND SECOND PAIR IDENTIFY RECEIVER
C                - (*,1) - FROM TYPE #
C                - (*,2) - FROM UNIT #
C                - (*,3) - TO TYPE #
C                - (*,4) - TO UNIT #
C   MAXCON - MAX NUMBER OF CONNECTIONS IN IADS STRUCTURE
C   NUMCON - # OF CONNECTIONS TO BE READIN IN IADS STRUCTURE DATA FILE
C   MAXIADSTGTS - MAX NUMBER OF IADS UNITS THAT ARE NON-SAM TARGETS
C   NUMIADSTGTS - TOTAL NUMBER OF IADS UNITS THAT ARE NON-SAM TARGETS
C              - IN THIS INPUT DATAFILE
C   IDIADSTGT(MAXIADSTGTS,3) - FOR EACH NON-SAM TARGET NUMBER, THIS
C         - HOLDS THE IADS TYPE AND UNIT NUMBER, USED TO IDENTIFY
C         - WHAT THE NON-SAM TARGET IS. (*,1) - NON-SAM TGT #;
C         - (*,2) - IADS TYPE #; (*,3) - IADS UNIT #
C   IADSTGT(10) - FLAG(1/0) INDICATING IF THIS NON-SAM TARGET
C         - IS A IADS TARGET OR NOT. =1 IS IADS TYPE TARGET. =0 NOT.
C         - USED TO INDENTIFY IS A NON-SAM TARGET IS AN IADS TARGET.
C         - (ARRAY MUST BE DIMENSIONED > = THAN NTARGETS)
C
      INTEGER CONNECT,MAXCON,NUMCON,NUMIADSTGTS,IDIADSTGT,IADSTGT,
     &MAXIADSTGTS
      PARAMETER(MAXCON=50)
      PARAMETER(MAXIADSTGTS=5)
```

```
      COMMON/IADSCON1/CONNECT(MAXCON,4),NUMCON,NUMIADSTGTS,
     &IDIADSTGT(MAXIADSTGTS,3),IADSTGT(10)
C
---------------------------------
C
C   PARAMETERS OF EWDATA DEF
C
C   EWPROCDLY - NORMAL PROCESSING DELAY FROM DETECTION OF PENETRATOR
C           - UNTIL EW SITE READY TO SEND MESSAGE (SECONDS)
C   EWDAMDLY(9) - MAX. EXTRA DELAY UNTIL EW SITE READY TO SEND MESSAGE
C             - DUE TO SITE BEING DAMAGED
C   EWDAMRAT(9) - SCALE FACTOR USED WITH EWDAMDLY(9) TO COMPUTE DELAY
C             - DUE TO SITE DAMAGE
C   EWPSMIN - MIN PS FOR EW SITES TO SAY IN SIMULATION (NOT KILLED)
C
      REAL EWPROCDLY,EWDAMDLY,EWDAMRAT,EWPSMIN

      COMMON/EWDATA/EWPROCDLY(9),EWDAMDLY(9),EWDAMRAT(9),EWPSMIN
C
---------------------------------
C
C   PARAMETERS OF EWMES2 DEF
C
C   REWMESID(9,MAXEWMESID)-
C     - HOLDS THE ID NUMBER OF EACH UNIQUE MESSAGE TO EACH EW SITE. USED
C     - TO COMPARE INCOMING MESSAGES AND REJECT PREVIOUSLY RECEIVED
C     - MESSAGES
C   MAXEWMESID - MAX SIZE OF RECEIVED ID ARRAY
C     REWMESIDPOS(9) - ARRAY POINTER WHERE NEXT MESSAGE ID# MAY BE
INSERTED
C               - FOR EACH EW SITE
C
      INTEGER MAXEWMESID,REWMESID,REWMESIDPOS
      PARAMETER(MAXEWMESID=50)

      COMMON/EWMES2/REWMESID(9,MAXEWMESID),REWMESIDPOS(9)
C
---------------------------------
C
C   PARAMETERS OF EW1 DEF
C
C   EWDETECT(K,IM) - FLAG(1/0) =1 IF EW SITE K CURRENTLY DETECTS PEN IM
C               - =0 EW SITE CURRENTLY DOES NOT DETECT PEN IM.
C   EWSTATS(9)  - STATUS(1/0) OF EW SITES. =1 RADIATING
C   START - FLAG(1/0) =1 INDICATES SIMULATION JUST STARTED. USED IN
C         - INITIALIZING SUBS
C   UPEWTGT(9,IM) - FLAG(-1/0/1) INDICATING THE NEWLY UPDATED DETECTION
C               - RESULTS OF EACH EW SITE. -1= LOST DETECTION, 0= NO
C               - CHANGE FROM LAST TIME, 1= NEW DETECTION
C
```

```
      INTEGER EWDETECT,EWSTATS,START,UPEWTGT

      COMMON/EW1/EWDETECT(9,IM),EWSTATS(9),START,UPEWTGT(9,IM)
C
---------------------------------
C
C  PARAMETERS OF HOLDM1 DEF
C
C MESHOLD(200,MAXMSGSIZE) - HOLDS MESSAGES GENERATED BY IADS ELEMENTS
C              - EACH ELEMENT IN NOW DESCRIBED
C              - (K,1) - TIME EVENT OCCURRED
C              - (K,2) - TIME WHEN MESSAGE IS TO BE SENT
C              - (K,3) - PS OF EW SITE WHEN EVENT OCCURRED
C              - (K,4) - SITE TYPE # (EW SITE = 1) WHERE MES IS FROM
C              - (K,5) - SITE ID # (I.E. EWID(K)) WHERE MES. IS FROM
C              - (K,6) - SITE TYPE # (EW SITE = 1) WHERE MES IS TO
C              - (K,7) - SITE ID # (I.E. EWID(K)) WHERE MES. IS TO
C              - (K,8) - ACTION # DESIRED WITH THIS MESSAGE
C              - (K,9) - # FOR UNIQUE MESSAGE IDENTIFICATION
C              - (K,10) - ORIGINATING IADS TYPE #
C              - (K,11) - ORIGINATING IADS UNIT #
C              - (K,12) - FINAL DESTINATION IADS TYPE #
C              - (K,13) - FINAL DESTINATION IADS UNIT #
C  NEXTHOLDPOS - POINTER INDICATING LOCATION OF NEXT AVAILABLE
C            - POSITION FOR INPUTTING A MESSAGE INTO THE ARRAY
C  EWDETMES(K,MAXMSGSIZE,MAXEWMES) -
C              - HOLDS DATA ASSOCIATED WITH A DETECTION MESSAGE FROM
C              - EW SITE # K. EACH ELEMENT IN NOW DESCRIBED
C              - (K,1) - TIME EVENT OCCURRED
C              - (K,2) - TIME WHEN MESSAGE IS TO BE SENT
C              - (K,3) - PS OF EW SITE WHEN EVENT OCCURRED
C              - (K,4) - SITE TYPE # (EW SITE = 1) WHERE MES IS FROM
C              - (K,5) - SITE ID # (I.E. EWID(K)) WHERE MES. IS FROM
C              - (K,6) - SITE TYPE # (EW SITE = 1) WHERE MES IS TO
C              - (K,7) - SITE ID # (I.E. EWID(K)) WHERE MES. IS TO
C              - (K,8) - ACTION # DESIRED WITH THIS MESSAGE
C              - (K,9) - # FOR UNIQUE MESSAGE IDENTIFICATION
C              - (K,10) - ORIGINATING IADS TYPE #
C              - (K,11) - ORIGINATING IADS UNIT #
C              - (K,12) - FINAL DESTINATION IADS TYPE #
C              - (K,13) - FINAL DESTINATION IADS UNIT #
C  EWDETDAT(9,IM,MAXEWMES) -
C               - HOLDS THE UPDATED DETECTION CHANGES ASSOCIATED WITH
C               - THE MESSAGE. FORMAT IS -1= LOST DETECTION, 0= NO
C               - CHANGE, 1= NEW DETECTION OF PENETRATOR IM
C  EWMESPOS(9) - INDEX IN MESSAGE HOLDING ARRAY TO INPUT NEXT MESSAGE
C            - FOR EACH EW SITE
C  MAXEWMES - MAX NUMBER OF MESSAGES IN HOLDING ARRAY
C  MAXMSGSIZE - MAX # OF DATA ELEMENTS IN MESSAGE
C
```

```
      INTEGER EWDETDAT,EWMESPOS,MAXEWMES,MAXMSGSIZE,NEXTHOLDPOS

      PARAMETER(MAXEWMES=20)
      PARAMETER(MAXMSGSIZE=13)

      REAL EWDETMES
      REAL MESHOLD(200,MAXMSGSIZE)

      COMMON/HOLDM1/EWDETMES(9,MAXMSGSIZE,MAXEWMES),
     & EWDETDAT(9,IM,MAXEWMES),EWMESPOS(9),MESHOLD,NEXTHOLDPOS

-----------------------------------
C
C   PARAMETERS OF KNOWN1 DEF
C
C DETTGT(9,IM) - INDICATES DETECTION STATUS FROM EACH EW SITE
C             - BASED ON RECEIVED MESSAGES. VALUE IS 1 IF NEW DETECTION
C             - =0 IF NO CHANGE AND =-1 IF LOST DETECTION ON THIS
C             - PENETRATOR
C DETALL(IM)   - INDICATES COMBINED DETECTION STATUS FROM ALL EW SITES
C             - BASED ON RECEIVED MESSAGES. VALUE IS # OF EW SITES
C             - THAT HAVE DETECTED THIS PENETRATOR. IF =0, PENETRATOR
C             - IS NOT DETECTED BY ANY SITES.
C
      INTEGER DETTGT,DETALL

      COMMON/KNON1/DETTGT(9,IM),DETALL(IM)
C
-----------------------------------
C
C   PARAMETERS OF MESID DEF
C
C   MSGID - NUMBER USED TO UNIQUELY IDENTIFY MESSAGES. USED TO TAG
C         - MESSAGES IN MESSAGE HOLDING ARRAY AND ASSOCIATED DETECTION
C         - STATUS DATA
C
      INTEGER MSGID

      COMMON/MESSID/MSGID
C
-----------------------------------
C
C   PARAMETERS OF QIADS DEF
C
C   NEWSITES - MAX # OF EW SITES
C   EWID - SAM ID # DESIGNATING THIS AS AN EW SITE
C   EWSITE - FLAG(1/0) =1 INDICATES THAT THIS SITE IS AN EW SITE
C
      INTEGER NEWSITES,EWID,EWSITE
```

```
      COMMON/QIADS/NEWSITES,EWID(9),EWSITE(JM)
C
-----------------------------------
C
C   PARAMETERS OF RPTDATA DEF
C
C  NUMRPT - MAX NUMBER OF EW REPORTING STATIONS IN THIS SIMULATION.
C  MINRPTPS - MIN. SURVIVAL PROB. FOR EW RPT STATION TO SAY IN SIM.
C  RPTPROCDLY - NORMAL PROCESSING DELAY FROM DETECTION OF PENETRATOR
C            - UNTIL EW REPORTING SITE READY TO SEND MESSAGE (SECONDS)
C  RPTDAMDLY(9) - MAX. EXTRA DELAY UNTIL EW REPORTING SITE READY TO
C               - SEND MESSAGE DUE TO SITE BEING DAMAGED
C  RPTDAMRAT(9) - SCALE FACTOR USED WITH RPTDAMDLY(9) TO COMPUTE DELAY
C               - DUE TO EW REPORTING SITE DAMAGE
C
      REAL RPTPROCDLY,RPTDAMDLY,RPTDAMRAT,MINRPTPS
      INTEGER NUMRPT

      COMMON/RPTDATA/RPTPROCDLY(9),RPTDAMDLY(9),RPTDAMRAT(9),
     & NUMRPT,MINRPTPS
C
-----------------------------------
C
C   PARAMETERS OF RPTMES1 DEF
C
C   RPTMESID(9,MAXRPTMESID)-
C    - HOLDS THE ID NUMBER OF EACH UNIQUE MESSAGE TO EACH EW REPORTING
C    - STATION. USED TO COMPARE INCOMING MESSAGES AND REJECT PREVIOUSLY
C    - RECEIVED MESSAGES
C   MAXRPTMESID - MAX SIZE OF RECEIVED ID ARRAY
C  RPTMESIDPOS(9) - ARRAY POINTER WHERE NEXT MESSAGE ID# MAY BE INSERTED
C                 - FOR EACH EW REPORTING STATION
C
      INTEGER MAXRPTMESID,RPTMESID,RPTMESIDPOS
      PARAMETER(MAXRPTMESID=50)

      COMMON/RPTMES1/RPTMESID(9,MAXRPTMESID),RPTMESIDPOS(9)
C
-----------------------------------
C
C   PARAMETERS OF SAMDATA DEF
C
C  SAMPROCDLY - NORMAL PROCESSING DELAY FROM DETECTION OF PENETRATOR
C            - UNTIL SAM SITE READY TO SEND MESSAGE (SECONDS)
C  SAMDAMDLY - MAX. EXTRA DELAY UNTIL SAMSITE READY TO SEND MESSAGE
C            - DUE TO SITE BEING DAMAGED
C  SAMDAMRAT - SCALE FACTOR USED WITH SAMDAMDLY(9) TO COMPUTE DELAY
C            - DUE TO SITE DAMAGE
C
      REAL SAMPROCDLY,SAMDAMDLY,SAMDAMRAT
```

```
      COMMON/SAMDATA/SAMPROCDLY,SAMDAMDLY,SAMDAMRAT
C
-----------------------------------
C
C   PARAMETERS OF SAMMES1 DEF
C
C   RSAMMESID(JM,MAXSAMMESID)-
C     - HOLDS THE ID NUMBER OF EACH UNIQUE MESSAGE TO EACH SAM SITE.
C    - USED TO COMPARE INCOMING MESSAGES AND REJECT PREVIOUSLY RECEIVED
C     - MESSAGES
C   MAXSAMMESID - MAX SIZE OF RECEIVED ID ARRAY
C   RSAMMESIDPOS(JM) - ARRAY POINTER WHERE NEXT MESSAGE ID# MAY BE
C                - INSERTED FOR EACH SAM SITE
C
      INTEGER MAXSAMMESID,RSAMMESID,RSAMMESIDPOS
      PARAMETER(MAXSAMMESID=50)

      COMMON/SAMMES1/RSAMMESID(JM,MAXSAMMESID),RSAMMESIDPOS(JM)
C
-----------------------------------
```

INITIAL DISTRIBUTION LIST

1.	Library, Code 52 Naval Postgraduate School Monterey, CA 93943-5000	2
2.	OR Academic Group, Code 30 Naval Postgraduate School Monterey, CA 93943-5000	1
3.	Prof. Ball Aeronautical Engineering Section Naval Postgraduate School Monterey, CA 93943-5000	1
4.	Mr. Gino LaMarca Weapons Planning Group, Code 304 Naval Weapons Center China Lake, CA 93555-6001	2
5.	Mr. Bryan F. Smith Weapons Planning Group, Code 304 Naval Weapons Center China Lake, CA 93555-6001	2
6.	LCRD William J. Walsh Code OR/WA Naval Postgraduate School Monterey, CA 93943-5000	1
7.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2

845-216

Thesis
S57385 Smith
c.1 A model of an Integrated Air Defense Systems (IADS) for the TACOPS program.

Thesis
S57385 Smith
c.1 A model of an Integrated Air Defense Systems (IADS) for the TACOPS program.

AMERICAN BINDERY 2 9-93

3 2768 00018470 9